

ISSN 0236-2554

ИНТЕРКОМПЬЮТЕР

INTERCOMPUTER

5/90

КОМПЬЮТЕРЫ НА БАЗЕ МИКРОПРОЦЕССОРА 80386

ПРАКТИЧЕСКИЕ СОВЕТЫ ПОЛЬЗОВАТЕЛЯМ PCAD

РЕЗИДЕНТНЫЕ ПРОГРАММЫ НА ЯЗЫКЕ СИ

ВЫЙДУТ ЛИ ФИРМЫ ZEOS INTERNATIONAL И ADOBE SYSTEMS НА СОВЕТСКИЙ РЫНОК?

RISC И UNIX

МОДЕЛЬ АНТИВИРУСНОЙ ПРОГРАММЫ-МОНИТОРА

ВОРОВСКАЯ УТОПИЯ В ВЕК ИНФОРМАТИКИ

КРИПТОГРАФИЯ И КОМПЬЮТЕРЫ

ПО ВОПРОСАМ ПОДПИСКИ

НА ЖУРНАЛ

ИНТЕРКОМПЬЮТЕР

INTERCOMPUTER

обращайтесь в ЦНТУ "Информ"

В 1991 г. будет выпущено шесть номеров. Стоимость подписки на 1 год 25 руб 80 коп. Количество экземпляров не ограничивается. Для осуществления подписки необходимо прислать в ЦНТУ "Информ" письмо с указанием необходимого количества экземпляров и номера платежного поручения о переводе денег на расчетный счет ЦНТУ "Информ". В адрес заказчика будет направлено гарантийное обязательство ЦНТУ.

Адрес ЦНТУ "Информ":
111141, Москва, Зеленый пр., 7
Телефон: 176-68-88
Расчетный счет 345034 в Коопбанке Центросоюза СССР, корреспондентский счет 161406 в ЦОУ при Госбанке СССР (МФО 229112).

ЦНТУ "Информ"
организует также рассылку журнала
"Интеркомпьютер"
NN 1, 2 за 1989 г.
и отдельных номеров за 1990 и 1991 гг.
наложенным платежом.



ИНТЕРКОМПЬЮТЕР INTERCOMPUTER 5/90

В НОМЕРЕ

Механизмы создания индустрии программного обеспечения. Свои соображения по этому поводу высказывает Б. Маркварт, управляющий фирмы Ashton-Tate **5** *Mechanisms that create a software industry. B. Marquart, business development manager for Ashton-Tate Europe states his opinion in this connection*

Компьютеризация по-советски и нормальные экономические отношения **8** *Computerization in a Soviet way and normal economic relations*

Несколько советов пользователям системы P-CAD **20** *Several pieces of advice for P-CAD users*

"Интеркомпьютер" начинает публикацию глав из книги "Руководство для покупателя компьютера 386" американского "технического писателя" Эдвина Ратча **22** *"Intercomputer" begins publication of chapters from the book "386 Computer Buyer's Guide and Handbook" written by American technical writer Edvin Rutsch*

Криптография и компьютеры. Что дает этот альянс? **28** *Cryptography and computers. What this union gives?*

Модель антивирусной программы-монитора **42** *Model of anti-virus monitor program*

СОДЕРЖАНИЕ

СОБЫТИЯ 3

Всемирный конгресс по компьютерному праву
С днем рождения, BYTE!
Первый звонок в Пилотной школе
Профессор Вирт в МГУ

МНЕНИЯ 5

От индустриального общества к информационному.
Б. Маркварт
Люди и компьютеры в экономическом зазеркалье.
А.Н. Козырев

ГОВОРЯТ УЧАСТНИКИ МЕЖДУНАРОДНОГО КОМПЬЮТЕРНОГО ФОРУМА 13, 31

"SOFT LETTER" О 100 ВЕДУЩИХ АМЕРИКАНСКИХ ФИРМАХ-РАЗРАБОТЧИКАХ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ПК 14

СЕМЕЙНЫЕ ХРОНИКИ 18

IBM: RISC и UNIX.
В.В. Леонас

ВОКРУГ САПР 20

Несколько советов пользователям P-CAD.
А.Ю. Кузнецов

ЛИКБЕЗ 22

Руководство для покупателя компьютера 386.
Э. Ратч
Криптография и компьютеры.
М.А. Богданов

БИБЛИОТЕЧКА ИНТЕРКОМПЬЮТЕРА 32

Turbo Basic.
Б.А. Ашкинази

АНТИВИРУСЫ 42

Антивирусная программа VP - Virus Protector.
В.Б. Комягин

КОПИЛКА ОПЫТА 48

Резидентные программы на языке Си.
Рекомендации разработчикам.
В.Э. Дембский

Учредитель - коллектив редакции журнала "Интеркомпьютер"
Главный редактор А.А. Эйдес
Заведующая редакцией О.В. Толкачева
Редактор М.С. Гордон
Главный художник А.С. Широков
Художественный редактор В.И. Солодкова
Фото А.А. Волкова, В.Н. Кирюхина
Компьютерный набор: М.М. Егорова, И.Ш. Карасик
Производство и распространение: В.А. Кромская
Адрес для переписки: 125190, Москва, А-190, а/я 240

Ярославский полиграфкомбинат Госкомпечати СССР
Зак. 1279 Объем 8,6 уч.-изд. л. 40 000 экз. Цена 3 руб.



На пять минут забудем о компьютерах

Редакция "Интеркомпьютера" хочет на время отвлечь своих читателей от дисплеев и поговорить с ними по душам: подвести некоторые итоги, поделиться проблемами.

"Интеркомпьютеру" исполнился год. Первый номер журнала вышел в ноябре 1989 г., а 5 ноября нынешнего года "Интеркомпьютер" влился в пеструю семью российских периодических изданий, получив "Свидетельство о регистрации средства массовой информации" под номером 315.

Итак, выпущены семь очередных номеров и один специальный к Международному компьютерному форуму. В этом году выйдет еще один, шестой, номер "Интеркомпьютера" (мы собирались выпустить еще седьмой и восьмой, но бумажный дефицит не позволил этого сделать).

Пусть не волнуются те подписчики, которые внесли плату за восемь номеров - в счет подписки на 1990 г. они получат два первых номера за следующий год. В дальнейшем мы решили остановиться на цифре 6: нам представляется, что при сегодняшнем состоянии рынка советских компьютерных журналов оптимальная периодичность выхода "Интеркомпьютера" - один раз в два месяца. Скептикам отвечаем, что нам, наконец, удалось решить проблемы с бумагой и в 1991 г. мы постараемся обеспечить обещанную ритмичность выхода журнала.

Условия подписки на "Интеркомпьютер" приведены на 2-й странице обложки. Внимательного читателя заинтересует, почему цена подписки несколько выше, чем в 1990 г., - отвечаем: розничная цена журнала не изменилась, возросли почтовые расходы.

Как бы там ни было, редакция "Интеркомпьютера" верит, что и в следующем году мы будем регулярно встречаться с читателями на страницах нашего журнала. Постоянные читатели уже поняли, что "Интеркомпьютер" - лучший компьютерный журнал; надеемся, к такому же выводу придут и те читатели, которые встретятся с этим журналом впервые...

P.S. К сожалению, любители полезных советов не найдут в этом номере ставшую уже привычной рубрику *Советы Бориса Алексеевича*: Борис Алексеевич в очередном отпуске.

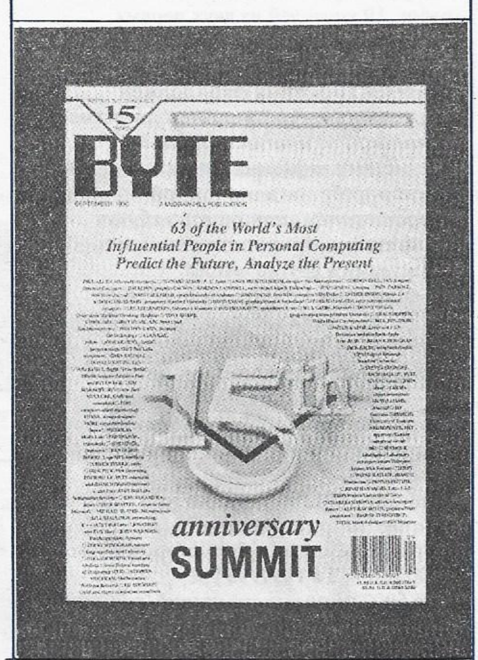
Ждите следующего номера!

ВСЕМИРНЫЙ КОНГРЕСС ПО КОМПЬЮТЕРНОМУ ПРАВУ

Компьютерный бизнес опутал земной шар сетями формальных и неформальных связей, для которых не существует ни государственных границ, ни естественных преград. Идет интенсивный обмен идеями, конструкциями, программами, стандартами. Как в таком случае быть с разночтениями в соответствующих законодательствах передовых стран Запада? Как быть с теми странами, в которых (как в СССР) подобные законы вообще отсутствуют?
Рассмотрению этих, а также многих других вопросов и посвящен Всемирный конгресс по компьютерному праву (World Computer Law Congress), который пройдет в Калифорнии (США) с 18 по 20 апреля 1991 г.
Основная задача конгресса - обмен мнениями, опытом и информацией.
В этом международном форуме примут участие около 500 юристов, бизнесменов, технических специалистов, представителей правительственных организаций из многих стран мира, в том числе и из Советского Союза.
В ходе конгресса будут вручены почетные награды: "Пионеру компьютерного права," "Лучшему юристу" и др.
Итак, до конгресса почти полгода. Может быть, и в нашей стране за это время будут наконец приняты законодательные акты, предусматривающие защиту прав авторов компьютерных программ.
Вот уже несколько лет разными инстанциями ведется подготовка соответствующих проектов, но ни один из них пока не снискал популярности у компьютерной общественности.
Что будет с советским компьютерным законодательством - покажет время, а пока стало известно, что доктор Ф. Абрамсон (знакомый постоянным читателям "Интеркомпьютера" по статье об экспорте программных продуктов) взялся помочь разработчикам программ из Европы выйти на американский рынок. Он и другие юристы из фирмы United Software Associates будут распространять в США конкурентоспособные программные продукты, а также регистрировать патенты, торговые марки и защищать авторские права.
Тот, кто хочет попытать счастья на американском рынке, может обратиться к Ф. Абрамсону по телефаксу: (301) 869-5636.
Желаем успехов!

А. Аба

С ДНЕМ РОЖДЕНИЯ, BYTE!



Исполнилось 15 лет журналу BYTE. Его первый номер вышел в свет в сентябре 1975 г. В следующем году начал выходить Dr. Dobb's Journal, позднее появились PC Magazine, PC World и многие другие журналы, но первым периодическим изданием по персональным компьютерам был BYTE. Собственно сам термин "personal computer" (персональный компьютер) впервые появился на страницах этого журнала, как и многие другие компьютерные термины: backslash, boot, clone, hacker, user, WYSIWYG..
BYTE - ровесник микрокомпьютерной индустрии: именно в 75-ом фирма MITS выпустила первый компьютер, который можно считать персональным, - Altair 8800. Сердцем этому компьютеру служил микропроцессор Intel 8080, а емкость его оперативной памяти равнялась 256 байтам.
Тот год был отмечен и другими важными для истории развития персональных компьютеров вехами. Фирма Zilog разработала микропроцессор Z80. Билл Гейтс (Bill Gates) и Поль Аллен (Poul Allen) завершили первую версию языка BASIC для микрокомпьютеров и создали фирму Microsoft.
Появился первый персональный компьютер (Sphere I) на основе микропроцессора Motorola 6800.
Компьютер Sphere I, в отличие от предшественников, внешне гораздо больше походил на современные ПК, был снабжен встроенными монитором и клавиатурой, оперативной памятью емкостью 4 Кбайт и мониторной системой, "зашитой" в ПЗУ...
Сегодня технические данные первых ПК выглядят смехотворно по сравнению с характеристиками современных моделей. Индустрия персональных компьютеров за 15 лет сделала небывалый скачок, старался не отставать от нее и BYTE. Журнал первым открыл на своих страницах лабораторию по тестированию компьютеров и программ (BYTE Lab), которая разработала первые наборы тестов (banchmarks), широко используемые сейчас многими специалистами во всем мире. Таким образом, читатель обрел в лице журнала не только надежного информатора, но и внимательного советчика - прочитав BYTE, всегда можно сделать правильный выбор необходимого устройства или программного продукта.
По мере распространения персональных компьютеров постоянно увеличивался тираж журнала. Сейчас BYTE - самое массовое компьютерное издание, его читают более полумиллиона людей во многих странах. Не исключено, что в недалеком будущем BYTE гостеприимно раскроет страницы и перед советскими читателями - во всяком случае, упорные слухи об этом ходят...
По подсчетам главного редактора журнала Фреда Ланги (Fred Langa), за 15 лет своего существования BYTE опубликовал около 150 Мбайт информации - более миллиарда бит.
"Интеркомпьютер" желает собратьям по перу, чтобы второй миллиард был столь же интересным и полезным для читателей, как и первый!

СОБЫТИЯ

ПЕРВЫЙ ЗВОНОК В ПИЛОТНОЙ ШКОЛЕ

Начало нового учебного года в московской школе N 1217 было необычным: 3 сентября здесь состоялось торжественное открытие первой Пилотной школы, проведенное в рамках советско-американского проекта "Pilot Schools" - "Пилотные школы" (очевидно, "пилотные" в смысле "задающие направление"). Этот

СОБЫТИЯ

долгосрочный проект, осуществляемый совместно Госкомитетом СССР по народному образованию и фирмой IBM, отражает переориентацию процесса обучения в школах на IBM-совместимые технические средства, в связи с чем предусматривает создание уже в ближайшие годы в нескольких тысячах советских школ компьютерных классов, оборудованных компьютерами семейства PS/2 фирмы IBM.

Выступая на церемонии открытия, руководитель проекта "Пилотные школы", первый заместитель председателя Госкомитета СССР по народному образованию Ф. Перегудов, в частности, сказал: "Сейчас время требует компьютеров нового, более высокого уровня и большей надежности. Я думаю, сотрудничество с фирмой IBM, при котором наши школьники получают возможность работать на самых современных персональных компьютерах (этих компьютеров до сих пор нет в других отраслях), может значительно ускорить процесс компьютеризации нашей страны". Вице-президент фирмы IBM Trade Development (Europe) Карл Таборски (Karl Taborsky) и другие представители фирмы IBM, участвовавшие в церемонии открытия, вручили первой группе учителей Пилотных школ сертификаты о прохождении ими соответствующей переподготовки.

В рамках проекта "Пилотные школы" фирма IBM взяла на себя поставку технических средств (часть из них - в качестве дара) и базового программного обеспечения, а также обучение советских специалистов.

С советской стороны для подготовки программно-методических

СОБЫТИЯ

материалов, координации эксперимента в Пилотных школах и анализа его результатов создан Учебный научно-методический центр новых информационных технологий (УНМЦ НИТ), входящий в состав акционерного социалистического предприятия "Компьютерный учебно-демонстрационный издательский сервисный центр" (АСП КУДИЦ).

Ряд сотрудников УНМЦ НИТ прошел подготовку в европейских подразделениях фирмы IBM, а в июне этого года уже в самом центре прошли переподготовку, завершившуюся защитой дипломных работ, 19 учителей из двух первых Пилотных школ (школы N 1217 и N 1234 г. Москвы). УНМЦ НИТ при участии учителей Пилотных школ должен подготовить программно-методические комплексы по многим школьным предметам для разных возрастных групп. После апробации и возможной корректировки эти комплексы будут поставляться вместе с техническими средствами в другие школы.

В I полугодии 1990/91 учебного года в компьютерных классах двух первых Пилотных школ будут проводиться уроки по информатике (8-е и 9-е классы), математике и английскому языку (7-е - 9-е классы), а начиная со II полугодия - и по другим предметам.

К началу 1991/92 учебного года компьютерами семейства PS/2 фирмы IBM должны быть оборудованы 1000 школ, а в 1992/93 учебном году - уже 5000 школ по всей стране (последнее, правда, при условии, что удастся найти необходимые для закупки этих компьютеров средства). Со своей стороны фирма IBM и связанные с ней банки предлагают Советскому Союзу помощь в изыскании этих средств (путем создания совместных предприятий, строительства и эксплуатации гостиниц и т.п.).

Проект "Пилотные школы" является одним из трех проектов, осуществляемых в настоящее время Госкомитетом СССР по народному образованию вместе с фирмой IBM. Эти проекты составляют часть проекта "Культура, наука и образование: США - СССР" (рассчитанного на 15 лет), соглашение о котором было подписано во время визита Президента СССР М.С. Горбачева в США в июне 1990 г.

А.Д. Плитман

ПРОФЕССОР ВИРТ В МГУ

25 сентября 1990 года в МГУ состоялась лекция Никлауса Вирта (Niklaus Wirth), профессора Технического университета г. Цюриха, автора всемирно известных языков Паскаль, Модула, Модула-2. Профессор Вирт рассказал о своей новой разработке - объектно-ориентированном языке Оберон.

"Поводом" для создания языка Оберон послужила начатая Виртом

СОБЫТИЯ

и его коллегой Гутнехтом (J. Gutknecht) работа над операционной системой для нового компьютера Ceres. Подобно тому, как совершенствование инструментальных средств при разработке вычислительной системы Lilith в свое время привело к появлению языка Модула-2, развитие этого языка при написании ядра операционной системы для компьютера Ceres вылилось в создание нового

СОБЫТИЯ

объектно-ориентированного языка.

Чтобы лучше понять предпосылки создания языка Оберон, необходимо сказать несколько слов о самом компьютере Ceres и его операционной системе. Профессиональный персональный компьютер Ceres базируется на 32-разрядном микропроцессоре NS32GX32, имеет цветной графический монитор с разрешением 1024х800 пикселей, жесткий диск емкостью 80 Мбайт и оперативную память емкостью от 4 до 8 Мбайт.

Операционная система для этого компьютера разрабатывалась как многооконная однопроцессная система с мощными средствами управления окнами и широкими возможностями расширения. Выполняющий функции базового языка системы Оберон получен путем введения в язык Модула-2 средств динамического порождения новых типов и наследования свойств. Это позволило заменить ряд конструкций языка на более общие (в частности, исключить интервальные типы) и несколько упростить синтаксис.

Не надо быть пророком, чтобы предсказать новому языку большое будущее. Написанный в расчете на обучение студентов элегантный Оберон сочетает в себе простоту синтаксиса и широту возможностей. Как утверждает сам Н. Вирт, появление языка Оберон знаменует новый шаг в программировании: от объектно-ориентированных систем к системам, ориентированным на расширение (extension-oriented).

Можно добавить, что некоторые фирмы уже приступили к разработке коммерческих компиляторов языка Оберон.

Подробнее о лекции Н. Вирта, языке Оберон и перспективах объектно-ориентированного подхода вы сможете прочитать в одном из ближайших номеров "Интеркомпьютера".

А.И. Масалович

МНЕНИЯ

Брюс МАРКВАРТ

Речь на Международном форуме по проблемам правовой охраны интеллектуальной собственности в области программирования, Переславль-Залесский, 29 июня 1989 г.

ОТ ИНДУСТРИАЛЬНОГО ОБЩЕСТВА К ИНФОРМАЦИОННОМУ

Я являюсь управляющим фирмы Ashton-Tate по коммерческой деятельности в Европе. Ashton-Tate - крупнейший поставщик программного обеспечения для персональных компьютеров. Штаб-квартира фирмы находится в Лос-Анджелесе (Калифорния), а ее филиалы расположены в 10 странах. Мы распространяем программное обеспечение на 10 языках, не считая английского, более чем в 70 странах.

Основная тема моего выступления - механизмы создания индустрии программного обеспечения. В марте 1988 г. я выступал в Центральном экономико-математическом институте АН СССР на симпозиуме, посвященном экспорту программного обеспечения. Я высказал тогда и сейчас повторяю свой основной тезис: "Необходим кодекс нравственных норм по использованию информации, причем особенно важно добровольное уважение правил лицензионных соглашений об использовании программного обеспечения персональных компьютеров".

Прежде всего позволю себе высказать некоторые положения общего характера, с которыми, как я полагаю, многие из вас знакомы и могут согласиться.

Путь от индустриального общества к информационному можно разделить на несколько этапов: от зарождения и роста рынка персональных компьютеров до их широкомасштабного использования миллионами пользователей. Каждый этап характеризуется особенностями поведения пользователей, разработчиков программного обеспечения и правительства. Можно получить огромные экономические выгоды, добиться повышения производительности труда и извлечь пользу для всего общества, действуя в соответствии с определенными правовыми и нравственными нормами, однако можно и не достичь всего этого, если эти нормы будут иными.

Примечательно, что каждому этапу присущи определенные ключевые события, отмечающие периоды перехода к достижению большей производительности и извлечению большей пользы для всего общества.

Отмечу некоторые особенности развития рынка, наблюдавшиеся в разных странах на различных этапах.

- Переход от одного уровня производительности к другому не всегда определяется числом приобретенных персональных компьютеров. Есть много примеров того, как большое количество денег может быть потрачено на персональные компьютеры впустую, особенно на начальных этапах быстрого роста их парка.
- Во всех организациях и социальных структурах глубоко укоренилось сопротивление каким бы то ни было изменениям. Это сопротивление всегда будет противостоять применению персональных компьютеров, так как оно основано на страхе - страхе от незнания того, как информация будет использоваться правительствами и программистами, страхе потери работы, страхе перед машинами, страхе перед изменениями вообще.
- До сих пор достаточно успешное развитие индустрии программного обеспечения (в основном на Западе) обеспечивалось только его решительной защитой, которая осуществлялась с помощью трех следующих механизмов:

1. Строгие законы об авторском праве.

Законы об авторском праве должны защищать каждый аспект разработки программного обеспечения и определять характер его использования. Некоторые формы использования должны быть ограничены или запрещены совсем. Например, право свободного доступа к "общественному фонду" программного обеспечения, предоставляемое всем желающим, должно распространяться на продукцию только тех разработчиков, которые сами хотят распространять свои программы таким образом.

2. Лицензионные соглашения, санкционирующие использование программного обеспечения.

Эти соглашения определяют пожелания группы авторов (разработчиков) относительно того, каким образом программное обеспечение может или не может быть использовано; отсюда термины "санкционированное использование" и "несанкционированное использование". Плата за пользование может сильно меняться в зависимости от намерений автора; цену устанавливает именно группа авторов (разработчиков), а не какая-либо высшая инстанция.

3. Применение законов об авторском праве и связанных с ними международных законов об охране интеллектуальной собственности в гражданском и уголовном судах.

КОПИРОВАНИЕ ПО ГРУППАМИ ПОЛЬЗОВАТЕЛЕЙ

Индустрия персональных компьютеров своим успехом во многом обязана распространению групп пользователей, осуществляющих практическую проверку программного и аппаратного обеспечения. Однако деятельность этих групп ни в коем случае не должна способствовать незаконному копированию ПО.

Несанкционированное копирование ПО группами пользователей делает положение многих людей уязвимым. Как и отдельные лица, незаконно копирующие ПО, группы пользователей и владельцы помещений, предоставляемых для проведения встреч их членов, должны нести ответственность за нарушение авторского права.

Тесная связь между группами пользователей и издателями ПО выгодна обеим сторонам. Группам пользователей следует поощрять корректное использование ПО своими членами. В свою очередь ассоциации издателей ПО должны убеждать своих членов выполнять требования пользователей по обеспечению соответствующих поддержки и модернизации ПО.

Все законы об авторском праве должны пройти практическую проверку в суде. Судьи, применяющие эти законы, должны иметь соответствующие познания о компьютерах и программном обеспечении. Иногда старые законы следует изменить и предусмотреть применение мер уголовного наказания за крупномасштабное копирование и распространение программ без разрешения авторов (разработчиков).

Эти три механизма могут создать экономические условия для развития инфраструктуры нового типа, обеспечивающей распространение знаний, услуг, программного обеспечения сначала на внутреннем рынке и затем (практически сразу же) за рубежом.

Затем эта инфраструктура разрастается и распространяется на более традиционные области деятельности, в особенности на производство, финансовую сферу и образование. Новая индустрия программного обеспечения помогает перестроить рабочие процессы с целью подъема производства, улучшения обслуживания, повышения эффективности распределения, уменьшения отходов и улучшения образования.

Однако, как мы все замечали в других областях жизни, изменения в мышлении и морали немногих передовых в некотором отношении людей (но не большой части общества) обычно происходят до изменений в национальном законодательстве. Всегда имеется естественный конфликт между старым и новым.

КАК ИЗБЕЖАТЬ "КРАЖИ" ПО В МЕСТАХ ОБЩЕЙ РАБОТЫ

В местах общей работы имеется две разновидности "кражи" ПО: изготовление копий ПО, которые служащие берут домой, и изготовление копий ПО для использования на рабочих местах. Обе ситуации означают, что приобретенных законным путем копий ПО недостаточно для нормального функционирования всех применяемых в процессе работы компьютеров.

При неправильных взаимоотношениях между издателями и пользователями ПО последним приходится следовать правилу "один пакет ПО - один компьютер". *Это означает, что число пакетов ПО должно соответствовать числу работающих компьютеров.* Покупая столько программ, сколько их может потребоваться, фирма устраняет стимул, побуждающий служащих к несанкционированному копированию. Придерживаясь этого правила, фирма расплачивается за ПО на длительное время, тогда как незаконно копируя ПО, она попадает в ужасную зависимость. Другой возможностью для фирмы является заключение с издателями ПО специальных лицензионных соглашений, позволяющих возместить издателям убытки, которые они могут понести при копировании ПО в соответствии с правилом "один пакет ПО - один компьютер". В то же время при этом устраняется возможность нарушения авторского права.

Большое число пакетов ПО продается вместе с локальными вычислительными сетями (ЛВС). Если Ваша фирма имеет ЛВС, обязательно следуйте руководствам издателей ПО при его использовании в ЛВС. Применение в ЛВС копии ПО, приобретенной в единственном экземпляре, для одновременной работы более чем одного пользователя является нарушением авторского права и большинства лицензионных соглашений.

Установлено, что в тех случаях, когда служащие фирмы подписывают заявление об использовании ПО, подтверждающее их намерение обеспечить защиту авторского права, кражи ПО происходят реже. Образец заявления, распространяемого Ассоциацией издателей ПО (Software Publishers Association - SPA), приведен во вставке на с. 7.

Вот четыре вывода, которые я сделал за годы работы:

- ☐ Без нового морального кодекса, регулирующего использование информации, страх людей перед изменениями, связанными с применением компьютеров, не может быть ослаблен или ликвидирован. Разногласия перерастают в конфликты, конфликты порождают реакционеров, реакционеры способствуют усилению страха, и круг замыкается.

Наибольшее опасение вызывает предположение, что правительства могут использовать информацию против граждан. На международном уровне этот страх проявляется в отношениях между правительствами, являющимися, по общему мнению, потенциальными военными противниками. Мы знаем, какой вред нам нанесли взаимные торговые эмбарго. И они сохраняются, пока существуют взаимное недоверие и боязнь перемен.

- ☐ Без объяснения людям, в особенности молодежи, того, что является "санкционированным и законным", а что - "несанкционированным и незаконным", невозможно широкомасштабное эффективное распространение знаний и практических навыков по использованию современного программного обеспечения.

Обращение к молодым людям особенно важно - в их умах заключено будущее. Однако они не рождаются со знанием новых концепций. Им нужно объяснить, каким образом соблюдение нравственных норм при использовании информации и уважение к интеллектуальной собственности могут принести обществу пользу.

- ☐ Без нового морального кодекса поведения невозможно создать рынок, способный собрать средства, необходимые для быстрого развития инфраструктуры нового типа и роста индустрии программного обеспечения.

КОПИРУЯ ПО, ВЫ СОВЕРШАЕТЕ КРАЖУ

Большинство людей не нарушают закон специально. Им никогда не придет в голову украсть деньги из чьего-либо кармана. Однако тот, кто копирует ПО без согласия автора, ворует интеллектуальную собственность и должен сознавать последствия своих действий.

Если Вы - индивидуальный пользователь, не нарушайте закон. Кто-то расплачивается за Ваше преступление. Если Вы работаете в организации, следите за тем, чтобы заявление о защите ПО было подписано и изложенные в нем требования не нарушались.

Все это приводит меня к следующему, самому главному выводу, полученному как путем теоретических размышлений, так и в результате моего опыта работы.

☐ Без нового международного морального кодекса поведения невозможно достичь долговременных экономических выгод в создаваемом информационном обществе, поскольку без него не могут успешно существовать крупномасштабный национальный рынок и индустрия программного обеспечения. Зерно не может прорасти на поле из камня.

Без успешного развития внутреннего рынка не будет и крупного конкурентоспособного рынка для экспорта программных продуктов.

Возможно, высказанные мною мысли кажутся негативными, но они не являются моим прогнозом на будущее, а опыт работы в Ashton-Tate не дает оснований для пессимизма.

Как же определить и предсказать моменты перехода от одного этапа развития производства к другому?

Я полагаю, что мы можем точнее всего определить вехи, отделяющие один этап развития производства от другого, как последовательность позитивных шагов на пути к утверждению единых международных норм в области авторского права на программное обеспечение и нравственных норм в использовании информации. Этими шагами являются, в частности, совместные встречи лидеров отрасли, например, такие, как наша сегодняшняя встреча.

Как показывает мой опыт, подобные события отмечали моменты перехода на пути к достижению большей производительности, обеспечиваемой применением персональных компьютеров.

В пределах отдельной страны возможна следующая последовательность продвижения по этому пути:

1. Разработчики программного обеспечения и ведущие деятели науки, экономики и производства делают все возможное, чтобы заставить правительство отказаться от поддержки "общественного фонда" программного обеспечения и защитить авторские права разработчиков программного обеспечения.

2. Персональные компьютеры в большом количестве поступают в школы и институты. Допускается разработка программ на языках всех национальных меньшинств, но официальный язык должен преподаваться в качестве образца. Затраты на обучение и приобретение лицензий на использование программного обеспечения для каждой машины достигают 50% стоимости установленного в учебных заведениях компьютерного оборудования.

Большое внимание следует уделять выбору и подготовке преподавателей, которые будут обучать молодежь. Здесь предстоит решить три основные задачи:

- Эффективное использование компьютеров и программного обеспечения в обучении.
- Соблюдение нравственных норм при получении доступа к информации и добровольное выполнение правил лицензионных соглашений об использовании программного обеспечения, установленных разработчиками.
- Обеспечение максимально возможного числа людей средствами, необходимыми для решения общественно полезных задач.

3. Рост национального рынка лицензий на программное обеспечение и развитие инфраструктуры, поддерживающей рынок. Появление возможности в течение 24 часов получить ответ на профессиональный запрос из любого места страны.

4. Приведение законов об охране авторского права в соответствие с международными стандартами. Введение новых судебных процедур; получение судьями (некоторые из них сами могут быть пользователями!) соответствующего образования в этой области. Правосудие должно защищать права как отечественных, так и зарубежных пользователей и разработчиков программ и наказывать тех, кто извлекает прибыль путем их нелегального копирования и распространения.

5. Экспорт программного обеспечения начинается с поставки на мировой рынок конкурентоспособных экспертных систем и прикладных программ в области медицины, экономики, литературы, математики, спорта, сельского хозяйства, производства, техники, музыки и т.д.

6. Разработчики и основные потребители программного обеспечения выступают за введение действенного "декрета о свободе" для избавления от страха перед возможностью использования компьютеров правительством против народа в репрессивных целях.

Описанные этапы развития не являются ни фантазией, ни планом, навязываемым могучей западной державой. Это схематическое изложение пятилетней программы развития системы образования в королевстве Иордания. Сотрудники фирмы Ashton-Tate удовлетворены тем, что внесли свою лепту в успешное выполнение этой программы.

ОБРАЗЕЦ ЗАЯВЛЕНИЯ ОБ ИСПОЛЬЗОВАНИИ ПО

1. Фирма (агентство) дает разрешение на использование ПО, разработанного различными сторонними фирмами. Фирма (агентство) не является владельцем этого ПО и относящейся к нему документации и не имеет права на его копирование без разрешения разработчика.

2. При работе в ЛВС или на нескольких машинах служащие фирмы (агентства) будут использовать ПО только в соответствии с лицензионным соглашением.

3. Служащие фирмы (агентства), знающие о любом злоупотреблении ПО или относящейся к нему документацией, известят об этом заведующего отделом или адвоката фирмы (агентства).

4. В соответствии с законом США об авторском праве в качестве наказания за незаконное копирование ПО могут применяться взыскание штрафа в размере до 50 000 дол и тюремное заключение. Служащие фирмы (агентства), изготавливающие, приобретающие или использующие копии ПО без разрешения на то авторов (разработчиков), будут наказаны в соответствии с обстоятельствами. Наказанием может быть увольнение со службы. Фирма (агентство) не прощает незаконного копирования ПО.

Я ознакомлен с заявлением об использовании ПО и поддерживаю его.

____

Подпись служащего и дата

Software Publishers Association
1101 Connecticut Ave., NW, Suite 901
Washington, D.C. 20036
202-452-1600

МНЕНИЯ



А.Н. КОЗЫРЕВ

ЛЮДИ И КОМПЬЮТЕРЫ В ЭКОНОМИЧЕСКОМ ЗАЗЕРКАЛЬЕ

Чтобы победить воровскую утопию "легкого хлеба", недостаточно ни созерцательного экстаза, ни парения над житейским, ни даже молитвенного подъема к святому и чудесному. Для этого нужно живое дело.

Е. Трубецкой

"Иное царство" и его искатели в русской народной сказке

Отношения, складывающиеся вокруг ПК, дают богатейший материал для очерка нравов и рассуждений на тему "Как относительно законным способом заработать миллион". Но статья не об этом, а о нормальных экономических отношениях, вырастающих из первозданного хаоса "компьютеризации по-советски".

Экономическим зазеркальем я называю не только теневую экономику, но и всю систему экономических отношений, основанных на подавлении деловой активности, внеэкономическом принуждении к труду и неуважении к творчеству. Строго говоря, это отнюдь не вся наша экономика и, может быть, даже не большая ее часть, но все, о чем будет сказано ниже, безусловно существует. Уже осужденное на слом и покинутое своими двуличными апологетами экономическое зазеркалье продолжает существовать на фоне перестройки, гласности и относительно массовой компьютеризации страны, преломляет компьютеризацию по-своему и порождает новые формы теневого бизнеса.

Легализация деловой активности в СССР в конце 80-х годов совпала по времени с массовым проникновением в страну персональных компьютеров (ПК). Поэтому не случайно ПК стал не только символом компьютеризации и информатизации общества, но и символом делового успеха, т.е. типичным предметом "демонстративного потребления", по терминологии Торстена Веблена (Т. Веблен. *Теория праздного класса.* М.: Прогресс, 1984).

Отношения, складывающиеся вокруг ПК, дают богатейший материал для очерка нравов на тему "Как относительно законным способом заработать миллион". Но статья не об этом, а о нормальных экономических отношениях, вырастающих из первозданного хаоса "компьютеризации по-советски".

Теневая экономика и компьютеризация

Существует традиционное деление теневой экономики на "черную" и "серую". К первой относятся уголовно наказуемые деловые операции, ко второй - вся экономическая деятельность, не подпадающая под статьи уголовного кодекса, но и не учитываемая официальной статистикой. Именно к этой разновидности теневого бизнеса относятся у нас в СССР несанкционированное копирование компьютерных программ, обмен готовым программным обеспечением (ПО) и многие другие операции, связанные с использованием вычислительной техники, лишь незначительная часть которых безоговорочно осуждается общественным мнением. Для "черной" экономики в компьютерном бизнесе пока еще нет достойного места, поскольку нет специального компьютерного законодательства. Есть косвенные свиде-

тельства проникновения в компьютерный бизнес организованной преступности, но в основном это обычный рэкет, контрабанда и другие не специфически "компьютерные" преступления.

В экономике дефицита любой достаточно ценный, легко транспортируемый и хорошо сохраняющийся продукт начинает выполнять некоторые функции всеобщего эквивалента. Не избежали этой участи и ПК, но только этим дело не ограничилось. В силу ряда обстоятельств ПК превратился в своеобразный символ преуспевания, как, например, импортный автомобиль, украшения и т.п. Фактически на ПК распространилось явление, названное Торстеном Вебленом "демонстративным потреблением" и описанное в его знаменитой книге *Теория праздного класса*.

Наличие "серой" экономики неизбежно, поскольку к ней относится вся не учтенная официально трудовая деятельность, но даже для вполне законопослушных граждан попадать в официальную статистику иногда слишком обременительно. Более того, многие законопослушные граждане не желают, чтобы их дополнительные заработки стали предметом обсуждения для соседей или коллег по работе. К сожалению, новый закон о налогообложении граждан, вступивший в действие 1 июля 1990 г., не учитывает этого обстоятельства. Вместе с тем вполне естественно стремление любого цивилизованного общества к возможно более полному отражению деловой жизни в официальной статистике, т.е. к минимизации "серой" экономики. Запретительные меры приводят, как правило, к расширению "черной" экономики за счет "серой" и "белой", т.е. легальной.

Разумеется, речь идет не о реальном росте преступности, происходит лишь ее формальный рост из-за расширения границ применения уголовного кодекса. Именно такого эффекта многие специалисты ожидают от включения ПО в число объектов, охраняемых с помощью авторского права. Основанием для подобных опасений является почти полная невозможность проконтролировать соблюдение соответствующих норм. Здесь, однако, следует учитывать, что введение правовой охраны ПО не сводется к запрету несанкционированного копирования. Важнейшим его следствием должно стать урегулирование на основе авторского права взаимоотношений между работодателями и программистами - авторами разрабатываемого ПО.

Очень существенной является возможность легализовать достаточно высокие доходы наиболее талантливых и удачливых программистов, вынужденных пока балансировать между "черным" и "серым" бизнесом. Оба эти фактора надо учитывать как способствующие снижению доли теневой экономики в компьютерном бизнесе. Зато безусловно вредны ограничения на торговую и посредническую деятельность, усиливаемые время от времени под предлогом борьбы со спекуляцией.

Сама по себе идея тотальной борьбы со спекуляцией порочна, поскольку спекулянт в экономике, по образному выражению С. Пачикова, подобен хищнику в природе и в той же мере ей необходим. Но дело не только в этом. На практике ограничения торговой деятельности приводят к искажению статистики и росту взяточничества, сдерживают ввоз в страну дефицитных товаров, в том числе вычислительной техники, но не создают сколько-нибудь серьезных препятствий для дельцов теневой экономики (вне зависимости от того, презираем мы их или превозносим). На индивидуальном ввозе ПК сколотили себе состояния тысячи наших соотечественников, но не меньшее их количество "погрело руки" на перепродаже ПК внутри страны именно благодаря наличию ограничений на ввоз.

Зыбкая грань между "серой" и "черной" экономикой все время перемещается под воздействием не очень умелого и весьма непоследовательного законотворчества. Чтобы не оказаться случайно в ее "черной" части, деловым людям необходимо и, как правило, достаточно соответствующим образом оформлять операции. Так, вместо договора о купле-продаже можно оформить договор о совместной производственной деятельности, вместо компьютеров продавать аппаратно-программные комплексы и т.д. В результате обычный дилер вынужден изображать из себя научно-техническую фирму, регистрировать ее и "создавать" необходимое ПО. Последнее можно делать, не обладая профессиональными навыками, не говоря уже о таланте, и не нарушая закона.

Дело в том, что советского покупателя обычно не заботят ни качество, ни лицензионная чистота ПО, приобретаемого вместе с компьютером, интересует только сам компьютер. Общий объем ПО, продаваемого у нас в СССР в нагрузку к ПК, оценивается в миллиарды, если не в десятки миллиардов рублей. Парадокс же состоит в том, что все это нагромождение нелепостей и лжи абсолютно безупречно с точки зрения существующего законодательства, тогда как деятельность честного дилера однозначно подпадает под статью уголовного кодекса.

Соблюсти интересы всех деловых партнеров и сохранить профессиональное самоуважение в создавшейся ситуации вполне возможно, если действительно поставлять каждый раз лицензионно чистое, высококачественное ПО. Это достойная, но очень трудная задача, постоянно осложняемая нетерпением партнеров, волокитой различных официальных служб и обычными человеческими слабостями. Самая распространенная из них - отсутствие достаточной квалификации при наличии большой амбиции и административных прав. Не менее серьезное препятствие на пути к нормальной торговле лицензионно чистым программным продуктом - компьютерное пиратство. Легкость получения незаконных копий любого сколько-нибудь распространенного программного продукта ставит потенциального покупателя в ложное положение, когда приобретение ПО на возмездной основе выглядит, в лучшем случае, благотворительностью, а в худшем - хищением в пользу третьего лица.

Компьютерное пиратство и борьба с ним

Несанкционированное копирование программ, или компьютерное пиратство, не является печальной привилегией СССР. Оно процветает во многих странах Европы и Азии, причем пиратские копии занимают тем большую долю в общем объеме используемого в стране ПО, чем слабее развита эта страна и чем менее глубоки в ней традиции соблюдения договорных обязательств. Эффективность судебного преследования за несанкционированное копирование программ достаточно низка практически во всех странах, поэтому правовую охрану ПО обязательно следует дополнять экономическими, техническими и моральными средствами защиты. Иными словами, воровать ПО должно быть экономически невыгодно, технически трудно и позорно. Всего этого можно добиваться как при наличии правовой охраны ПО, так и в ее отсутствие.

Технические и моральные средства борьбы с компьютерным пиратством имеет смысл обсуждать вместе, поскольку применение технических средств защиты означает открытую борьбу с предполагаемым грабителем. Здесь неизбежно возникает вопрос о допустимых пределах обороны, который сейчас встал особенно остро в связи с появлением компьютерных вирусов и их использованием для защиты ПО. Я считаю, что эту проблему каждый программист должен решать для себя сам, поскольку нигде в мире нет эффективных средств для судебного преследования компьютерных пиратов, как нет их и для преследования слишком активных борцов с компьютерным пиратством, не останавливающихся перед распространением компьютерных вирусов.

Безоговорочно следует осудить создание вирусов в целях самоутверждения, которое не имеет ничего общего с самообороной. Разрушительная энергия всеобщего стремления к самоутверждению столь велика, что уже сейчас начинают перевешивать те многочисленные плюсы, которые позволили этому стремлению стать одной из основных движущих сил современной научно-технической революции. Случай с компьютерными вирусами - яркая тому иллюстрация.

Чисто моральным средством борьбы с компьютерным пиратством может стать современная разновидность либерального террора, направленного против нарушителей профессиональной этики и кастовых интересов программистской элиты. Именно элита объективно заинтересована в пресечении компьютерного пиратства и придании программированию статуса творческой профессии подобно музыке, профессиональным шахматам и т.д. Но даже в нынешней ситуации, когда такого статуса у программирования нет, существует достаточно стимулов для формирования стереотипа негативной реакции на компьютерное пиратство и объединения на этой основе в своеобразную касту. Принадлежность к элите сама по себе достаточно притягательна, чтобы служить основой для такого объединения, если более элементарные потребности (заработок) достаточно легко удовлетворяются. Разумеется, легче обеспечить себе высокий заработок, если не считаться с моральными нормами и общественным мнением, но эта легкость быстро обесценивается с ростом квалификации.

Оценивая ситуацию в нашей стране в целом, берусь утверждать, что в настоящий момент программисты высшей квалификации морально готовы для объединения и коллективной борьбы с компьютерным пиратством, - вопрос только за умением найти друг друга и договориться. Для этого даже нет необходимости встречаться и вести прямые переговоры, - нужны поступки, т.е. ярко выраженная нега-

тивная реакция на каждый конкретный случай пиратства и своевременная информация об этом.

Разумеется, следует отдавать себе отчет в том, что такое поведение требует от программиста известного гражданского мужества и не всегда будет правильно понято окружающими, скорее наоборот. Синдром "поротой задницы", выработанный за годы безраздельного господства тоталитарного режима и подсказывающий советскому человеку, что лучше помалкивать, когда грабят тебя самого, и уж обязательно - если грабят соседа, пока еще не преодолен.

Приведу пример. На конференции по школьной информатике в Одессе (сентябрь 1989 г.) аспирантка Харьковского пединститута И. Забродa обнаружила свою программу с заставкой Талды-Курганского пединститута, о чем сразу же во всеуслышанье заявила. Дальше события развивались скучно, но поучительно. Практически вся почтенная публика посчитала, что не стоит поднимать скандала, поскольку "доказать ничего невозможно". Мое предложение организовать публикацию краткого сообщения по факту пиратства поддержала только потерпевшая, проявив незаурядную настойчивость в отстаивании своего права на имя. Все неподдержавшие ссылались на неизбежность ответной реакции со стороны предполагаемых пиратов. Столь же осторожную позицию заняли и знакомые мне сотрудники компьютерных журналов, ссылаясь на невозможность в таких случаях что-либо доказать.

Хитрость, однако, в том, что для профессионалов ситуации с пиратством практически всегда очевидны, ни о каком доказательстве, как и о судебном преследовании, не может быть и речи. Просто следует оповестить достаточно узкий круг заинтересованных лиц о каждом обнаруженном факте хищения; остальное, как говорится, приложится.

Экономические методы борьбы с компьютерным пиратством основываются на взаимной заинтересованности партнеров в долгосрочном сотрудничестве. Если речь идет о взаимоотношениях между разработчиками ПО и пользователями, то сотрудничество может заключаться в сопровождении программ, своевременной информации о найденных ошибках и улучшениях, предложении новых версий программного продукта со скидкой.

Возможны довольно экстравагантные шаги по налаживанию такого сотрудничества, например амнистия несанкционированных пользователей. Суть мероприятия состоит в принципиальном отказе разработчика мстить за совершенные ранее кражи программного продукта так называемым "несанкционированным пользователям", т.е. всем, кто, не имея на то санкции разработчика, использует данный программный продукт, но не занимается его перепродажей. По отношению к торговцам краденым ПО амнистии быть не может. Всем же несанкционированным пользователям предлагается обнаружить себя и представить доказательства реальной работы с продуктом, после чего они получают новую версию программного продукта и становятся постоянными клиентами фирмы. Экономическая целесообразность столь щедрого жеста несомненна, если фирма действительно способна постоянно совершенствовать и сопровождать свой продукт, но пока вынуждена это доказывать.

Дополнительная польза от амнистии - вытесние с рынка ПО недобросовестных конкурентов, т.е. торговцев краденым ПО. По мере накопления фирмой морального капитала и портфеля заказов амнистия должна становиться все более редким мероприятием. Одновременно возрастает вероятность карательных акций против пиратов. Первая в СССР амнистия несанкционированных пользователей осуществлена совместным предприятием "ПараГраф" в июле 1990 г. (амнистия несанкционированных пользователей старых версий системы "Бета"), о других случаях мне пока не известно.

Помимо сотрудничества разработчиков ПО и конечных пользователей совершенно необходимо долговременное сотрудничество между распространителями ПО. Речь идет прежде всего об организации лицензионной торговли программным продуктом, т.е. о создании в СССР цивилизованного рынка ПО. Вопреки широко распространенному мнению наличие правовой охраны программного продукта для этого вовсе не обязательно, скорее оно крайне желательно, так как позволяет строить отношения партнеров на ясной правовой основе. В любом случае гарантией соблюдения правил лицензионной торговли остается деловая порядочность партнеров, а не страх судебного преследования.

Существенным элементом лицензионной торговли является четко отлаженная процедура заключения договоров, внешне напоминающая ритуал с большим числом необязательных элементов. За каждым из них есть не всегда очевидный смысл, как и за многочисленными пунктами лицензионных договоров. Не случайно договор подписывает сначала покупатель (лицензиат), а потом продавец (лицензиар); программный продукт поставляется после перечисления паушального платежа. В конечном счете все эти "мелочи" обеспечивают слаженность экономического механизма и минимизируют возможность несанкционированного копирования ПО, поскольку программный продукт распространяется по уже готовой сети, "сотканной" из взаимных договорных обязательств.

Воровать программное обеспечение должно быть экономически невыгодно, технически трудно и позорно.

Похвальное слово ритуалам

Процедурные вопросы, возникающие при заключении лицензионных договоров о распространении ПО и в ходе их выполнения, заслуживают подробного рассмотрения. Во-первых, необходимо учитывать взаимную дополнительность таких товаров, как программные продукты, магнитные носители информации и компьютеры. Ведь ПО почти всегда продается вместе с магнитным носителем, а часто и вместе с компьютером. Это столь естественно и удобно, что не требует каких-либо доводов в обоснование. Но ситуация не так проста, как кажется на первый взгляд, и содержит в себе целый веер возможностей, о которых имеет смысл рассказать. Во-вторых, организация и функционирование сети сбыта на основе взаимных договорных обязательств требуют не только доверия между партнерами, но и предсказуемости их поведения, которая обеспечивается строгим соблюдением ритуалов.

Функционирование сети сбыта ПО удобно представить в виде трех отдельных тактов:

1. *Распространение прав на тиражирование и использование программного продукта.* Движение прав идет от автора к обладателю исключительной лицензии и далее через обладателей простых лицензий и сублицензий к конечному пользователю - покупателю этикеточной лицензии. В принципе, любой из промежуточных этапов, кроме отчуждения определенной части имущественных прав от автора и получения этикеточной лицензии конечным пользователем, может быть исключен.

2. *Движение денег от покупателей к автору, т.е. в обратном направлении.*

3. *Движение самого программного продукта от автора к конечным пользователям.*

При такой схеме минимизируется вероятность "растаскивания" программного продукта после начала его продажи, так как к моменту реального появления продукта в руках у первых пользователей достаточно ясно просматривается вся сеть договорных обязательств. Эта сеть имеет иерархическую структуру, причем для каждого уровня иерархии необходим свой тип договора; впрочем, все они похожи, поскольку в каждом речь идет о передаче прав, меняются лишь их объем и, возможно, какие-то сопутствующие условия.

Если продажа лицензии поддерживается поставкой магнитных носителей (дискет), то естественно разделить во времени этапы поставки магнитных носителей и самого продукта, поскольку плата за материальные ценности обычно осуществляется сразу после их получения или непосредственно перед этим. Иными словами, нельзя разрывать во времени получение денег и поставку дискет, но можно разорвать поставку пустых дискет и предоставление дистрибутива, т.е. дискеты с записью программного продукта, поставляемого его разработчиком для изготовления соответствующего числа копий. Разумеется, такой порядок действует только в середине описанной выше цепочки.

Автор передает исключительное право на распространение своего продукта, не предоставляя дискет для его распространения, а конечному пользователю продукт поставляется прямо на дискете. Соответствующим образом оформляются все лицензионные соглашения. В авторском договоре обязанности по тиражированию продукта и материальному обеспечению производственного процесса берет на себя покупатель; в лицензионных договорах эти обязательства могут быть распределены между партнерами произвольно.

Лицензионный договор сначала подписывается лицензиатом, а потом лицензиаром, что может быть достаточно важным, если подпи-

сание договора сторонами происходит неодновременно. Затягивание процедуры подписания договора лицензиаром достаточно безболезненно для лицензиата, поскольку в промежуточный период он может вести себя примерно так же, как и до подписания договора. При обратном порядке подписания договора возможно затягивание процедуры лицензиатом, в результате чего лицензиар оказывается в достаточно неудобном положении, так как он фактически вынужден отказывать другим потенциальным лицензиатам в продаже лицензий на распространение того же продукта.

Когда подобного рода мелочи войдут в ритуал, отпадет необходимость объяснять в каждом случае мотивы отказа от неудобных сделок и появится возможность поручить заключение договоров чисто техническим сотрудникам и т.п. В совокупности именно мелочи и составляют культуру торговли. Предсказуемость поведения - одна из основных характеристик, создающих образ солидной фирмы. Непредсказуемость в мелочах вызывает раздражение и желание никогда больше не связываться. В этом смысле самыми ненадежными партнерами являются государственные предприятия, руководителей которых все время куда-то срочно вызывают, поэтому в заранее оговоренное время их не оказывается на месте. На этот факт не принято обращать внимание, но аналогичное поведение для кооператора равносильно самоубийству.

“Конвертируемые” программисты и эксплуатация таланта

Эксплуатация человеческого труда и таланта занимает ключевое место среди причин, порождающих теневую экономику в наиболее распространенных и массовых ее формах. Не мифическим стремлением к сверхдоходам, а вполне естественным нежеланием подвергаться чрезмерной эксплуатации со стороны государства объясняется уход в “теневые” структуры многих талантливых людей в 70-е годы и в первой половине 80-х годов. Об этом свидетельствует печальная практика стран, где так или иначе подавлялась деловая активность.

Наша страна не являлась здесь исключением, ее опыт в основном лишь подтверждает общую тенденцию, хотя именно в СССР деловая активность подавлялась особенно последовательно и жестоко. Специфика страны проявилась главным образом в том, что с особым усердием подавлялись свободомыслие и свободная творческая деятельность; эксплуатация таланта с самого начала приняла гораздо более изощренные и жестокие формы, чем эксплуатация неквалифицированного труда. Жестокость эксплуатации проявилась не в физическом насилии или лишении элементарных средств к существованию, а в использовании гораздо более изощренных и разнообразных приемов воздействия на эксплуатируемых. Представление о том, как это делалось, можно получить читая “В круге первом” А. Солженицына.

Впрочем, далеко не обо всем из описанного выше следует говорить в прошедшем времени. Дух бериевских “шарашек” с их строгой иерархией пайков и мелких поблажек вместо соблюдения законных прав свободных личностей продолжает витать над нашей наукой, в том числе и академической. При соприкосновении с западным научным бытом он становится ощутимым и даже трудноперносимым, несмотря на нашу многолетнюю привычку.

В программировании, как и в других интеллектуальных занятиях, талант и квалификация значат неизмеримо больше, чем количество часов, проведенных программистом на рабочем месте, или официально занимаемая им должность. Объективная оценка таланта в высшей степени затруднительна. Поэтому все разговоры о равной оплате за равный труд и о благодеяниях, которыми начальство осыпает наиболее работоспособных программистов, стоят не больше, чем любые другие заклинания. Признание заслуг и таланта не всегда сопровождается оплатой, адекватной притязаниям, и т.д. Отсюда следует неизбежность конфликтов, принимающих новые, причем все более острые формы, в связи с открывающимися возможностями работать в иностранных фирмах при сохранении советского гражданства и “основного” места работы.

Конфликт, о котором пойдет речь ниже, достаточно хорошо известен научной и программистской общественности благодаря открытому письму С.Г. Воробьева директору Института программных систем АН СССР А.К. Айламазяну, которое было размножено и распространено его автором. В этом конфликте я полностью на стороне взбунтовавшегося программиста (С.Г. Воробьева), чего не собираюсь скрывать, но суть конфликта глубже личных симпатий и антипатий. В его основе - очевидные пороки ныне действующего в СССР трудового законодательства и всей системы найма, правовая незащищенность программиста и чудовищные диспропорции в оценке квалифицированного труда “у нас” и “у них”.

Старший научный сотрудник Института программных систем (ИПС) АН СССР С.Г. Воробьев был командирован в г.Бремен (ФРГ), где за 300 марок в неделю работал над совершенствованием разработанного при его активном участии в ИПС АН СССР информационно-ориентированного языка СТАРСЕТ. Для рядового советского программиста 300 марок в неделю - почти фантастическая зарплата, поскольку за 1500 марок можно купить хорошо отремонтированную подержанную “Ладу”, за 6 марок - самый дешевый плейер. Рядом на тех же условиях работали советские коллеги и были счастливы. Однако С.Г. Воробьев считал, что его труд стоит по крайней мере в пять, а то и в десять раз дороже, причем имел для этого все основания, поскольку мог сравнить свою квалификацию и зарплату с квалификацией и зарплатой западногерманских коллег. Так, программист, разрабатывающий для фирмы SIEMENS прикладную базу данных с многооконным интерфейсом на основе пакета INFORMAX, получал 1500 марок в день, имея совершенно свободный режим и не будучи даже обязан предоставлять исходные тексты программ. Наши программисты должны были сидеть на рабочем месте “от и до”, не имели возможностей покупать и читать книги. Кроме того, С.Г. Воробьев считал и считает, судя по распространенному им “открытому письму”, что имеет право, как автор, на все написанные лично им тексты программ и на все свои идеи, заложенные в язык СТАРСЕТ, тем более - на свои же идеи, не использованные ни в одной разработке института. Границы этих прав не зафиксированы ни в одном документе, поскольку вся работа строилась на обещании начальства, формулируемого емкой фразой “Не обидим!”. Контракта с программистом по работе над языком СТАРСЕТ ни институт, ни его западногерманский партнер не заключали, а контракт между ИПС АН СССР и западногерманской фирмой С.Г. Воробьеву даже не показали. Из неодинакового понимания сторонами справедливости оплаты за вложенный труд и возник конфликт, в результате которого С.Г. Воробьев, прервав командировку, вернулся в СССР, директор института А.К. Айламазян лишний раз съездил в ФРГ и объявил о возбуждении уголовного дела в связи с нанесением С.Г. Воробьевым ущерба государству в особо крупных размерах, а рядовые сотрудники института рискуют потерять возможность работы в ФРГ за 300 марок в неделю, т.е. практически за одни командировочные, причем для большинства из них это самое настоящее крушение надежд.

В данной истории поучительного не меньше, чем скандального. В основном это касается вопросов найма, поскольку существующая в нашей стране система трудовых договоров, трудовых соглашений и договоров подряда не способна отражать всю сложность взаимоотношений администрации с сотрудником, в работе которого может и должен присутствовать элемент творчества.

Подготавливаемая для использования в науке контрактная система должна быть более гибкой и богатой возможностями. В частности, необходимо предусмотреть возможности для разделения имущественных и личных неимущественных прав на научно-техническую продукцию между автором и нанявшей его организацией. В первую очередь это следует сделать для разработчиков ПО, поскольку в этой области уже сложились определенные международные нормы и не так уж сложно распространить их на нашу страну.

Для лиц, работающих за рубежом или в СССР по контрактам с иностранными фирмами, следует выработать хотя бы ориентировочные нормы оплаты труда. Прежде всего это должны быть ограничения снизу, т.е. гарантия некоторой минимальной оплаты, чтобы наши программисты не оказывались в положении граждан четвертого сорта. Возможно, что высокая оплата в конвертируемой валюте окажется уделом узкого слоя программистов, которых я бы условно назвал “конвертируемыми”, т.е. способными заменять в аналогичных условиях западных программистов. Их у нас не так уж много, по разным оценкам - от нескольких человек до нескольких тысяч, но пока еще они есть, хотя “неконвертируемых” гораздо больше.

Не случайно возможность работать за 10% нормальной "капиталистической" зарплаты представляется большинству наших программистов "манной небесной". По-видимому, надо дать им такую возможность. Однако работа на условиях "Не обидим!" сильно пахнет ранним капитализмом и представляет собой прорыв в светлое будущее разве что из крепостного права, именуемого у нас казарменным социализмом.

Воровская утопия и тоталитарное сознание

Реализацией воровской утопии назвал философ Е. Трубецкой созданное большевиками необычное общество, целиком ориентированное на распределение. В чистом виде это общество просуществовало недолго - с октября 1917 г. до введения новой экономической политики, но в гибридных и стертых формах проявило удивительную живучесть. Генетическому родству воровской утопии и Командно-административной Системы (далее просто Системы), по-видимому, еще будут посвящены специальные исследования, хотя особой тайны тут нет и сейчас. Основная опора Системы - маргиналы, или, по другой терминологии, люмпены (в широком смысле слова), т.е. люди, не вполне соответствующие занимаемому в обществе месту. Сила Системы проявляется в том, что она сама плодит маргиналов, призывая кухарок управлять государством и перекрывая талантливым людям всякую возможность прямого контакта с потенциальными покупателями их труда, способностей и результатов творчества.

Идейная основа воровской утопии - надежда беспомощного индивида на внешние силы и обстоятельства, которые могут принимать облик всемогущего государства, партии или директора института. Желание не прогневить становится преобладающим, а естественные для сильной и талантливой личности требования самостоятельности и уважения воспринимаются как крамола и неблагодарность. Истинные дети Системы - неквалифицированные и не обладающие необходимым для творчества талантом исполнители чужих указаний во всех областях человеческой деятельности, разумеется, не исключая программирования. Отсюда парадоксальное на первый взгляд явление - протест против эксплуатации со стороны именно тех людей, кому "страна дала все", т.е. чей талант и квалификация официально признаны обществом. В одном ряду оказываются известный детский писатель Э. Успенский, знаменитый футболист О. Блохин, шахматные "короли" Б. Спасский и Г. Каспаров. Теперь этот ряд пополняется программистами. Но если труд писателей и профессиональных шахматистов всегда признавался творческим, охранялись их авторские права и за ними даже признавалось право на часть их собственных валютных гонораров и призов, то для программистов все это - еще не пройденные "минные поля", где каждый неосторожный рывок вперед может стать, как минимум, концом карьеры. Пример С. Воробьева пока подтверждает это предположение.

Тоталитарное сознание отказывает программистам даже в праве на имя. В официальных отчетах и даже в публикациях имя автора программы обычно заменяется названием организации, в которой он работает. Грешит этим и "Интеркомпьютер", поместивший во втором номере за 1990 г. статью о текстовых редакторах с упоминанием организаций, но без указания фамилий реальных авторов: Е. Веселова, А. Борковского и Ю. Нестерова.

Следующая сложность для программиста - потребность в более дорогом, чем у писателя или шахматиста, инструменте, а иногда и необходимость иметь помощников. Кроме того, писателям помогает налаженная система издательств, выпускающих книги, художественные журналы и т.д. Ничего похожего, предназначенного для реализации программ, нет и, возможно, не может быть. Следовательно, программисту нужно самому думать об организации сбыта либо отказаться от самостоятельности.

Наиболее естественная для квалифицированного программиста форма организации труда - собственная фирма, в которой он выполняет основную работу, т.е. играет роль Мастера, а наемные служащие - вспомогательную, включая организацию сбыта, сопровождение программ и т.д. Такую фирму мы будем в дальнейшем называть личной. Личная фирма может распространять всего лишь один или два продукта, добиваясь их постоянного совершенствования и большого тиража.

Практика свидетельствует о быстром нарастании организационных сложностей с ростом числа наименований программных продуктов, хотя теоретически число этих наименований не ограничено. Идеологическая установка фирмы на одного лидера и один основной продукт означает ориентацию всех сотрудников на Мастера в ущерб собственным творческим амбициям и даже, может быть, в ущерб собственному таланту, поэтому идеализировать организацию производственного процесса в личных фирмах не стоит. Но это единственный вариант, при котором Мастер не подчинен Администратору и действительно занимает в фирме господствующее положение. Фирмы такого типа, несомненно, появятся после принятия правительственного постановления о малых предприятиях; фактически они уже есть, но вынуждены маскироваться под кооперативы. Для этого Мастер приглашает в "кооператив" двух пенсионеров и сразу же берет с каждого из них заявление о выходе из кооператива без указания даты. Пенсионеры получают за свою "службу" небольшие денежные пособия, но в случае разногласий с Мастером немедленно исключаются из кооператива на основании заранее написанных заявлений.

Существует не для всех очевидное сходство между воровской утопией и Командно-административной Системой, состоящее в принципиальном отказе от экономического принуждения к труду. В сказочном варианте воровской утопии ликвидация экономического принуждения к труду компенсируется волшебством. Наиболее обнаженно эта идея представлена в сказке "По щучьему велению". Реализация воровской утопии чудесами не сопровождалась, поэтому возникла необходимость во внеэкономическом принуждении к труду, которая не исчезла и по сей день.

Казарменный социализм - одна из стадий развития реализовавшейся воровской утопии. Вопреки мнению, распространенному среди демократической интеллигенции прошлого века и сохранившему живучесть до настоящего времени, эксплуатация не исчезает с уничтожением частной собственности на средства производства - исчезают сначала экономическая, а затем и интеллектуальная свобода, объективно мешающая укреплению тоталитаризма. В числе первых жертв интеллектуальной несвободы неизбежно оказывается наука, изучающая теневую экономику и порождающие ее дефекты Системы. Именно поэтому на протяжении нескольких десятилетий наша научная печать и публицистика даже не упоминали о теневой экономике, хотя само явление существовало все эти годы и играло в жизни нашего общества весьма заметную роль. Отсюда неадекватное восприятие ситуации массовым сознанием, связывающим появление теневой экономики с перестройкой, кооперативами и компьютеризацией. Между тем именно тоталитаризм до бессмыслицы исказил статистику, создав практически неограниченное поле деятельности для теневой экономики.

Бескомпромиссность в борьбе с теневой экономикой вполне может обернуться ее небывалым разгулом и другими бедами.

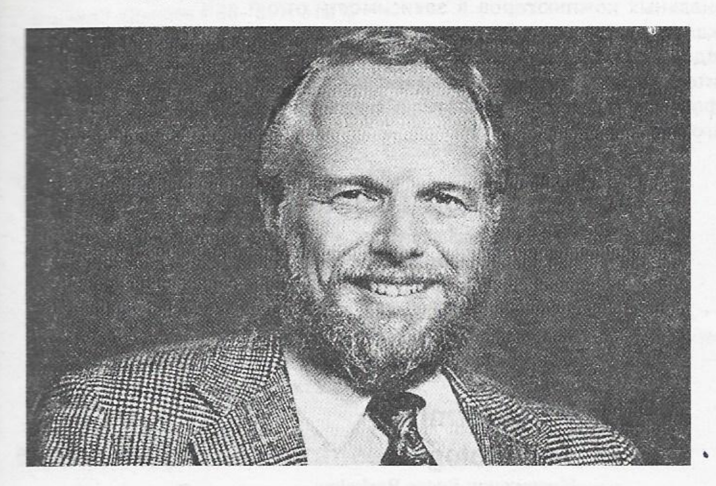
Крушение тоталитаризма не менее драматично. Продолжают действовать стереотипы тоталитарного мышления и адекватного ему поведения. В трудовом и гражданском законодательстве, в многочисленных подзаконных и противозаконных актах сохраняются рудименты Командно-административной Системы.

Несоответствие официального права интуитивному чувству справедливости и стремление значительной части власть предержащих быстрее обратить обесценивающиеся административные права в имущественные - два мощных источника, питающих теневую экономику. Компьютеризация - лишь фон, на котором разворачиваются события, хотя ее символ - персональный компьютер - оказался очень подходящим объектом для всякого рода спекуляций.

Тоталитарное сознание, не ведающее терпимости и компромиссов, в столь сложной ситуации становится особенно опасным. Бескомпромиссность в борьбе с теневой экономикой вполне может обернуться ее небывалым разгулом и другими бедами, подобно тому, как порыв русской интеллигенции к демократии и справедливости обернулся реализацией воровской утопии, а стремление к эффективности политической власти - возникновением в СССР и Германии невиданных по мощи и оголтелости тоталитарных режимов.

Поэтому хочется закончить в духе услышанной когда-то католической молитвы: "Боже, дай нам силы, чтобы преодолеть непереносимое, дай терпение, чтобы перенести непреодолимое, дай разум, чтобы отличить первое от второго".

ГОВОРЯТ УЧАСТНИКИ МЕЖДУНАРОДНОГО КОМПЬЮТЕРНОГО ФОРУМА



Фирма Adobe Systems (США), основанная в 1982 г., получила известность как разработчик языка Си, позволяющего сочетать текстовую и графическую информацию, выводимую на различные устройства печати. В настоящее время она входит в десятку ведущих американских фирм, разрабатывающих программное обеспечение для персональных компьютеров. Новая программа фирмы Adobe Type Manager позволяет получать четкое изображение на экране и высокое качество печати на принтере. Программа Adobe Photoshop может применяться для обработки как черно-белых, так и цветных изображений. Пакет Adobe Type Library представляет интерес для тех, кому нужны шрифты различных гарнитур. Программа Adobe Streamline позволяет преобразовывать информацию из побитового представления в файлы на языке PostScript.

Наш корреспондент взял интервью у президента фирмы Adobe Systems Джона Уарнока (John Warnock).

- Что привело Вас на МКФ?

Джон Уарнок: Я думаю, все на Западе с большим вниманием следят за изменениями в Советском Союзе, любые новости об этом имеют огромное значение. Мы очень рады, что наконец произошли сдвиги, открылись двери, изменяется политическая ситуация и начала изменяться экономическая. Я не сомневаюсь, что многие на Западе хотели бы помочь этому процессу, "стать хорошими соседями". Недавно Эстер Дайсон (Esther Dyson, президент фирмы EDventure Holdings, США) организовала конференцию, в которой участвовала и делегация из Советского Союза. Я тоже встречался с этой делегацией и именно тогда получил приглашение приехать на МКФ. Я с радостью принял приглашение, поскольку полагал, что смогу внести и свой вклад в этот процесс сближения.

- Насколько я понимаю, до приезда на МКФ у Вас уже сложилось какое-то представление о советском рынке. Изменилось ли оно после Вашего приезда в Советский Союз?

Д.У.: Изменилось, но не слишком. Еще очень много проблем предстоит решить. Очевидно, что проблема неконвертируемости рубля - одна из самых сложных, так же, как и проблемы, возникающие в связи с необходимостью принятия закона об охране интеллектуальной собственности. Однако все изменяется очень быстро - мы это видим и полны надежд на то, что нам удастся наладить деловые отношения.

- Каковы Ваши конкретные планы, связанные с советским рынком?

Д.У.: В настоящее время мы заинтересованы в том, чтобы иметь как можно больше информации о советском рынке. Конкретных планов, касающихся каких-либо определенных продуктов, у нас нет. Просто хотим побольше узнать о рынке, оценить свои возможности, "понаблюдать на близком расстоянии", если можно так выразиться.

The 1990 Soft•letter 100

"Интеркомпьютер" завершает публикацию сведений о том, как в 1990 г. распределились места между 100 ведущими американскими фирмами-разработчиками программного обеспечения для персональных компьютеров в зависимости от роста доходов (объема продаж) в 1989 календарном году. Эти сведения с разрешением на их публикацию любезно предоставил Международному компьютерному клубу Джеффри Тартер (Jeffrey Tarter), редактор и издатель американского журнала Soft•letter, дважды в месяц информирующего своих читателей о тенденциях и стратегии в издании программного обеспечения.

51 TimeWorks $9 800 000

Президент: Mark L. Goldberg
Число служащих: 72 — Год основания: 1982
Рост доходов ('88-'89): 3%
1988: $10 100 000 — 1987: $9 400 000
1986: $8 100 000 — 1985: $6 700 000
1984: $4 400 000 — 1983: $2 300 000
Категории продуктов: 2,3,7
Ориентация разработок: a,c,f,g,i

52 Best Programs $9 790 000

Президент: James F. Petersen
Число служащих: 108 — Год основания: 1982
Рост доходов ('88-'89): 26% — Место в 1989 г.: 55
1988: $7 800 000 — 1987: $5 014 000
1986: $3 017 000 — 1985: $1 777 000
1984: $1 266 000 — 1983: $265 000
Категории продуктов: 1,8,9
Ориентация разработок: a

53 The Learning Company $8 569 000

Президент: William Dinsmore
Число служащих: 60 — Год основания: 1979
Рост доходов ('88-'89): 8%
1988: $7 961 000 — 1987: $5 982 000
1986: $3 200 000
Категория продуктов: 3
Ориентация разработок: a,c,f

54 Software Shop Systems $8 500 000

Президент: Ron J. Koval
Число служащих: 90 — Год основания: 1978
Рост доходов ('88-'89): 8% — Место в 1989 г.: 46
1988: $9 200 000 — 1987: $8 250 000
1986: $6 700 000 — 1985: $5 300 000
Категории продуктов: 8,9
Ориентация разработок: a

54 Systems Plus $8 500 000

Президент: Richard W. Mehrlich
Число служащих: 31 — Год основания: 1980
Место в 1989 г.: 59
1988: $6 000 000 — 1987: $4 200 000
Категории продуктов: 8,9
Ориентация разработок: a,e

56 Revelation Technologies $8 390 000

Президент: Steve Perry
Число служащих: 105 — Год основания: 1983
Рост доходов ('88-'89): -8%
1988: $9 142 000
Категория продуктов: 6
Ориентация разработок: a,d

57 Datastorm Technologies $7 749 000

Президент: Bruce Barkelew
Число служащих: 43 — Год основания: 1985
Рост доходов ('88-'89): 81% — Место в 1989 г.: 71
1988: $4 280 000 — 1987: $1 734 000
1986: $325 000 — 1985: $3 000
Категория продуктов: 10
Ориентация разработок: a

58 Elcomp Systems $7 700 000

Президент: Ami Elis
Число служащих: 90 — Год основания: 1978
Рост доходов ('88-'89): 22%
1988: $6 300 000 — 1987: $4 500 000
1986: $3 200 000 — 1985: $2 800 000
1984: $2 300 000 — 1983: $1 800 000
Категории продуктов: 8,9
Ориентация разработок: a

59 Strategic Simulations $7 600 000

Президент: Joel A. Billings
Число служащих: 64 — Год основания: 1979
Рост доходов ('88-'89): 13% — Место в 1989 г.: 56
1988: $6 750 000 — 1987: S5 500 000
1986: $4 000 000 — 1985: $4 100 000
1984: $3 100 000 — 1983: S2 000 000
Категория продуктов: 2
Ориентация разработок: a,c,f,g,h,i

60 American Small Business Computers $7 200 000

Президент: Bruce Taylor
Число служащих: 25 — Год основания: 1980
Рост доходов ('88-'89): 33% — Место в 1989 г.: 54
1988: $5 414 000 — 1987: $4 000 000
1986: $3 000 000 — 1985: $2 500 000
Категория продуктов: 5
Ориентация разработок: a,c,d

Категории продуктов: 1 - коммерческие; 2 -развлекательные; 3 - образовательные; 4 - операционные системы/среды; 5 - САПР/графика; 6 - инструментальные средства/языки программирования; 7 - утилиты/шрифты; 8 - промышленные специального назначения; 9 - вычислительные; 10 - коммуникационные.
Ориентация разработок: a - DOS; b - Windows; c - Macintosh; d - OS/2; e - Unix; f - Apple II; g - Commodore 64/128; h - Amiga; i - Atari.

The 1990 Soft•letter 100

61 cc:MAIL — $7 000 000
Президент: Philippe Courtot
Число служащих: 35 — Год основания: 1983
Рост доходов ('88-'89): 312%
1988: $1 700 000 — 1987: $1 200 000
1986: $400 000
Категория продуктов: 10
Ориентация разработок: a,b,c,d

62 Persoft — $6 546 000
Президент: Ed Harris
Число служащих: 60 — Год основания: 1982
Рост доходов ('88-'89): 11% — Место в 1989 г.: 60
1988: $5 895 000 — 1987: $4 673 000
1986: $3 371 000 — 1985: $2 228 000
1984: $1 650 000 — 1983: $448 000
Категория продуктов: 10
Ориентация разработок: a,c,d

63 Automated Insurance Rating Services — $6 500 000
Президент: Donald Landgraf
Число служащих: 90 — Год основания: 1982
Рост доходов ('88-'89): 18%
1988: $5 500 000 — 1987: $5 000 000
Категория продуктов: 8
Ориентация разработок: a

64 California Software Products — $6 400 000
Президент: Peter Warkenton
Число служащих: 75 — Год основания: 1975
Рост доходов ('88-'89): -28% — Место в 1989 г.: 50
1988: $8 900 000 — 1987: $5 800 000
1986: $5 600 000 — 1985: $4 800 000
1984: $4 400 000 — 1983: $1 100 000
Категория продуктов: 1
Ориентация разработок: a,d

65 Data Access Corp. — $6 266 000
Вице-президент: C.L. Casanave III
Число служащих: 80 — Год основания: 1976
Рост доходов ('88-'89): -4% — Место в 1989 г.: 57
1988: $6 500 000 — 1987: $5 000 000
1986: $3 600 000 — 1985: $2 300 000
1984: $2 000 000 — 1983: $1 000 000
Категории продуктов: 1,6
Ориентация разработок: a,d

66 IMSI — $6 143 000
Президент: Geoffrey Koblick
Число служащих: 24 — Год основания: 1983
Рост доходов ('88-'89): 18% — Место в 1989 г.: 58
1988: $5 200 000 — 1987: $4 200 000
1986: $3 600 000 — 1985: $2 500 000
1984: $1 500 000 — 1983: $1 150 000
Категория продуктов: 5
Ориентация разработок: a

67 SoftDisk Publishing — $6 105 000
Президент: Al Vekovius
Число служащих: 95 — Год основания: 1981
Рост доходов ('88-'89): 109%
1988: $2 922 000 — 1987: $1 795 000
1986: $1 164 000 — 1985: $892 000
1984: $547 000
Категории продуктов: 1,2,3,7,9
Ориентация разработок: a,c,f,g

68 Matrix Software Technology — $6 100 000
Президент: Vincent Garofalo
Число служащих: 45 — Год основания: 1984
Рост доходов ('88-'89): 455%
1988: $1 100 000 — 1987: $375 000
1986: $150 000
Категории продуктов: 6,7
Ориентация разработок: a

69 Acius — $6 000 000
Президент: Marlene Delbourg-Delphis
Число служащих: 35 — Год основания: 1987
Рост доходов ('88-'89): 43%
1988: $4 200 000
Категория продуктов: 1
Ориентация разработок: c

70 Executive Systems/ Xtree Co. — $5 800 000
Президент: King R. Lee
Число служащих: 32 — Год основания: 1978
Рост доходов ('88-'89): 14%
1988: $5 100 000 — 1987: $1 700 000
Категории продуктов: 1,4,7
Ориентация разработок: a,c

71 Queue — $5 400 000
Президент: Jonathan D. Kantrowitz
Число служащих: 40 — Год основания: 1980
Рост доходов ('88-'89): 29% — Место в 1989 г.: 75
1988: $4 200 000 — 1987: $2 544 000
1986: $2 222 000 — 1985: $1 884 000
1984: $1 721 000 — 1983: $1 234 000
Категория продуктов: 3
Ориентация разработок: a,c,f,h

72 Reference Software — $5 243 000
Президент: Donald R. Emery, Ph.D.
Число служащих: 43 — Год основания: 1985
Рост доходов ('88-'89): 150%
1988: $2 100 000 — 1987: $1 035 000
1986: $300 000
Категории продуктов: 1,3,7
Ориентация разработок: a,c

73 Mathsoft — $5 048 000
Президент: David A. Blohm
Число служащих: 45 — Год основания: 1984
Рост доходов ('88-'89): 64% — Место в 1989 г.: 88
1988: $3 074 000 — 1987: $2 935 000
1984: $477 000
Категории продуктов: 1,3,8
Ориентация разработок: a,c,e

74 Media Cybernetics $4 600 000

Президент: Stephen Hall
Число служащих: 58 Год основания: 1981
Рост доходов ('88-'89): -8% Место в 1989 г.: 64
1988: $5 000 000 1987: $4 500 000
1986: $3 300 000 1985: $2 400 000
1984: $1 600 000 1983: $500 000
Категория продуктов: 6
Ориентация разработок: a,c,d,e

75 Individual Software $4 400 000

Президент: Jo-L Hendrickson
Число служащих: 40 Год основания: 1981
Рост доходов ('88-'89): 0% Место в 1989 г.: 72
1988: $4 400 000 1987: $3 000 000
1986: $2 100 000 1985: $1 800 000
1984: $1 500 000 1983: $1 000 000
Категории продуктов: 2,3,7
Ориентация разработок: a,c

76 Origin Systems $4 400 000

Президент: Robert K. Garriott
Число служащих: 36 Год основания: 1983
Рост доходов ('88-'89): -21% Место в 1989 г.: 65
1988: $5 600 000 1987: $3 800 000
1986: $3 600 000 1985: $3 000
1984: $1 500 000
Категория продуктов:-2
Ориентация разработок: a,c,f,g,h,i

77 Rightsoft $4 117 000

Президент: Pat DePree
Число служащих: 24 Год основания: 1983
Рост доходов ('88-'89): 30% Место в 1989 г.: 86
1988: $3 175 000 1987: $1 595 000
1986: $777 000 1985: $220 000
Категории продуктов: 1,3,7
Ориентация разработок: a,e

78 MapInfo $4 100 000

Президент: Michael D. Marvin
Число служащих: 51 Год основания: 1985
Рост доходов ('88-'89): 64% Место в 1989 г.: 100
1988: $2 500 000 1987: $1 060 000
1986: $103 000
Категории продуктов: 1,5,6,7
Ориентация разработок: a

79 Micro Planning Int. $4 000 000

Президент: Timothy A. Breen
Число служащих: 50 Год основания: 1978
Рост доходов ('88-'89): 33% Место в 1989 г.: 92
1988: $3 000 000 1987: $2 000 000
1986: $1 000 000 1985: $650 000
1984: $600 000 1983: $500 000
Категория продуктов: 1
Ориентация разработок: a,b,c,d,e

79 Wordtecl Systems $4 000 000

Президент: David B. Miller
Число служащих: 50 Год основания: 1980
Рост доходов ('88-'89): -18% Место в 1989 г.: 68
1988: $4 850 000 1987: $3 900 000
1986: $2 000 000 1985: $2 300 000
1984: $900 000 1983: $200 000
Категория продуктов: 1
Ориентация разработок: a,d,e

81 Gold Hill Computer $3 700 000

Президент: Gerald Barber
Число служащих: 28 Год основания: 1984
Рост доходов ('88-'89): -33%
1988: $5 500 000 1987: $7 700 000
1986: $2 500 000
Категории продуктов: 4,6
Ориентация разработок: a,b,c,e

82 Softsync $3 684 000

Президент: Ken Currier
Число служащих: 22 Год основания: 1981
Рост доходов ('88-'89): -3% Место в 1989 г.: 77
1988: $3 800 000 1987: $5 788 000
1986: $4 344 000 1985: $2 772 000
1984: S1 452 000 1983: $1 000 000
Категории продуктов: 1,9
Ориентация разработок: a,c

83 Libra Corp. $3 665 000

Президент: Willam A. Maasberg, Jr.
Число служащих: 57 Год основания: 1975
Рост доходов ('88-'89): 13% Место в 1989 г.: 74
1988: $3 237 000 1987: $3 224 000
1986: $3 762 000 1985: $4 387 000
1984: $4 647 000 1983: $3 780 000
Категории продуктов: 1,8,9
Ориентация разработок: a

84 Olduvai Corp. $3 575 000

Президент: Juan Chotsourian
Число служащих: 25 Год основания: 1987
Рост доходов ('88-'89): 238%
1988: $1 057 000 1987: $353 000
Категории продуктов: 1,7
Ориентация разработок: b,c

85 T/Maker Company $3 550 000

Президент: Heidi Roizen
Число служащих: 32 Год основания: 1979
Рост доходов ('88-'89): 0% Место в 1989 г.: 78
1988: $3 535 000 1987: $3 100 000
1986: $2 200 000 1985: $1 600 000
1984: $1 200 000 1983: $600 000
Категории продуктов: 1,5,7
Ориентация разработок: a,b,c,d,e

86 Computer Aided Management $3 500 000

Президент: Joseph Rizzo
Число служащих: 27 Год основания: 1984
Рост доходов ('88-'89): 40% Место в 1989 г.: 90
1988: $2 500 000 1987: $1 400 000
1986: $650 000 1985: $450 000
1984: $400 000
Категория продуктов: 1
Ориентация разработок: a

87 Chronos $3 400 000

Президент: Evin Ollinger
Число служащих: 15 Год основания: 1986
Рост доходов ('88-'89): 6% Место в 1989 г.: 83
1988: $3 200 000
Категория продуктов: 1
Ориентация разработок: a,b,d,e

The 1990 Soft•letter 100

87 Cosmi Corp. $3 400 000

Президент: George E. Johnson
Число служащих: 19 Год основания: 1982
Рост доходов (88-'89): 4% Место в 1989 г.: 82
1988: $3 265 000 1987: $2 547 000
1986: $2 168 000 1985: $1 747 000
1984: $1 132 000 1983: $626 000
Категория продуктов: 2
Ориентация разработок: a,g

89 Concentric Data Systems $3 350 000

Президент: John J. Henderson
Число служащих: 19 Год основания: 1979
Рост доходов ('88-'89): 22% Место в 1989 г.: 95
1988: $2 750 000 1987: $2 041 000
1986: $900 000
Категории продуктов: 1,7
Ориентация разработок: a,e

90 Jandel Scientific $3 338 000

Президент: John Osborn
Число служащих: 32 Год основания: 1982
Рост доходов ('88-'89): 8% Место в 1989 г.: 87
1988: $3 089 000 1987: $2 080 000
1986: $800 000 1985: $400 000
Категория продуктов: 8
Ориентация разработок: a,c

91 Blyth Software $3 262 000

Президент: Joseph J. Elkins
Число служащих: 50 Год основания: 1983
Рост доходов ('88-'89): 24% Место в 1989 г.: 99
1988: $2 628 000 1987: $4 542 000
Категории продуктов: 1,2,8,9
Ориентация разработок: a,b,c

92 Champion Business Systems $3 256 000

Президент: Scott Robertson
Число служащих: 45 Год основания: 1981
Рост доходов ('88-'89): 20%
1988: $2 707 000 1987: $2 021 000
1986: $1 508 000
Категории продуктов: 1,9
Ориентация продуктов: a,c

93 Berkeley Softworks $3 256 000

Президент: Brian P. Dougherty
Число служащих: 35 Год основания: 1984
Рост доходов ('88-'89): -35% Место в 1989 г.: 67
1988: $4 996 000 1987: $6 558 000
1986: $4 534 000 1985: $479 000
1984: $417 000
Категории продуктов: 1,2,4,6,7
Ориентация разработок: f,g

94 Chang Laboratories $3 200 000

Президент: Dash J. Chang
Число служащих: 20 Год основания: 1981
Рост доходов ('88-'89): 7% Место в 1989 г.: 89
1988: $3 000 000 1987: $1 500 000
1986: $4 561 000 1985: $6 237 000
1984: $2 659 000 1983: $2 754 000
Категории продуктов: 1,9
Ориентация разработок: c

95 Multisoft Corp. $3 180 000

Президент: Bruce Schafer
Число служащих: 20 Год основания: 1982
Рост доходов ('88-'89): 427%
1988: $603 000 1987: $749 000
1986: $127 000 1985: $178 000
1984: $65 000
Категория продуктов: 7
Ориентация разработок: a

96 Algor Interactive Systems $3 100 000

Президент: Michael Bussler
Число служащих: 36 Год основания: 1977
Рост доходов ('88-'89): 24%
1988: $2 500 000 1987: $1 200 000
Категории продуктов: 5,8
Ориентация разработок: a,e

96 Great American Software $3 100 000

Президент: Roger Melanson
Число служащих: 40 Год основания: 1984
Рост доходов ('88-'89): 15% Место в 1989 г.: 97
1988: $2 700 000 1987: $2 300 000
1986: $1 500 000
Категории продуктов: 7,9
Ориентация разработок: a

96 The Softa Group $3 100 000

Президент: Robert Grosshandler
Число служащих: 48 Год основания: 1982
Рост доходов ('88-'89): 11% Место в 1989 г.: 80
1988: $2 800 000 1987: $3 000 000
1986: $2 500 000 1985: $2 000 000
1984: $400 000 1983: $400 000
Категории продуктов: 8,9
Ориентация разработок: a,d

99 Enertronics Research $3 000 000

Президент: Douglas Wang
Число служащих: 18 Год основания: 1981
Рост доходов ('88-'89): 0% Место в 1989 г.: 91
1988: $3 000 000 1987: $2 500 000
1986: $2 500 000
Категории продуктов: 1,2,3,5,7
Ориентация разработок: a,b,d,e

99 Zenographics $3 000 000

Президент: Robert E. Romney
Число служащих: 34 Год основания: 1979
Рост доходов ('88-'89) 0%
1988: $3 000 000 1987: $2 500 000
1986: $1 000 000 1985: $1 000 000
Категории продуктов: 1,5,7
Ориентация разработок: a,b,d

СЕМЕЙНЫЕ ХРОНИКИ

В. В. ЛЕОНАС





IBM: RISC и UNIX

Появление компьютеров семейства RS/6000 - это по существу новый виток в конкуренции между ведущими фирмами, производящими средства вычислительной техники.

Корпорация IBM объявила, наконец, о выпуске долгожданного семейства RS/6000 (RISC System/6000), в состав которого входят девять моделей (четыре сервера и пять высокопроизводительных рабочих станций), являющихся по своему технологическому уровню вторым поколением систем с архитектурой типа RISC, т.е. RISC-систем. Эти системы, построенные на микросхемах є высокой степенью интеграции, изготовленных по КМОП-технологии, позволяют параллельно (одновременно) выполнять до 4-5 операций за один такт. Семейство RS/6000 - это совместная разработка двух исследовательских подразделений фирмы IBM, расположенных в Йорктауне (шт. Нью-Йорк) и Остине (шт. Техас).

Все модели семейства RS/6000 функционируют под управлением разработанной корпорацией IBM операционной системы (ОС) AIX, совместимой с широко известной ОС UNIX. Информация о некоторых моделях этого семейства представлена в табл. 1.

По сравнению с рабочими станциями фирм Sun Microsystems и DEC рабочие станции семейства RS/6000 примерно при тех же ценах имеют в 2-6 раз более высокую производительность (как в миллионах операций с фиксированной запятой в секунду - Mips, так и в миллионах операций с плавающей запятой в секунду - Mflops). Производительность модели 320 - младшей модели этого семейства - вдвое выше производительности рабочей станции Sun SPARCstation 1 и втрое выше производительности рабочих станций DEC 3100 и рабочих станций, построенных на базе микропроцессора Intel 80486. Старшая модель этого семейства - модель 540 - рассчитана на поддержку одновременной работы до 256 пользователей. По производительности модель 520, относящаяся ко второму поколению RISC-систем, превосходит модель RT6150, относящуюся к первому поколению этих систем, в 5 раз при измерении производительности в миллионах операций с фиксированной запятой в секунду и в 13 раз при измерении этой характеристики в миллионах операций с плавающей запятой в секунду. Результаты тестирования моделей семейства RS/6000 и ряда известных компьютеров других фирм, опубликованные в апрельском номере журнала "System International" за 1990 г., приведены в табл. 2.

Архитектура семейства RS/6000 ориентирована на 32-разрядную адресацию, что позволяет адресовать до 4 Гбайт физической памяти, а при использовании 52-разрядного виртуального адреса - до $4x10^6$ Гбайт виртуального адресного пространства. Шины данных между ОЗУ и кеш-памятью имеют разрядность 64 или 128 бит (в зависимости от модели), а скорость обмена информацией по этой шине - соответственно 160 или 480 Мбит/с. В компьютерах семейства RS/6000 в качестве системной шины используется усовершенствованная шина MCA, обеспечивающая пропускную способность до 40 Мбит/с в "импульсном" (burst) режиме (что вдвое выше пропускной способности шины MCA

Таблица 1

Модель	Конструктив	Производительность Mips	Mflops	Частота ЦП, МГц/ кеш, Кбайт	Число гнезд для ОЗУ	Объем ОЗУ, Мбайт базовый	макс.*	Число гнезд для MCA	Число гнезд для ВЗУ**	Базовый объем ВЗУ, Мбайт	Цена,*** дол.
320	Desktop	27.5	7.4	20/32	2	8	32/128	4	0/2	120	12.995
520	Deskside	27.5	7.4	20/32	8	8	128/512	7	3/6	355	27.245
530	Deskside	34.4	10.9	25/64	8	16	128/512	7	3/6	355	42.705
730	Deskside	34.5	10.9	25/64	8	16	128/512	6	3/6	355	41.250
930	Rack-mount	34.5	10.9	25/64	8	16	128/512	6	4/8	670	62.230
540	Deskside	41.1	13.0	30/64	8	64	128/512	7	3/6	640	92.885

Примечания:

* Числитель отношения представляет собой максимальную емкость ОЗУ при использовании микросхем памяти емкостью 1 Мбит, а знаменатель - ту же величину при использовании микросхем памяти емкостью 4 Мбит.

** Число гнезд для внешних ЗУ указано в виде отношения, числитель которого равен числу гнезд высотой 1/2, а знаменатель - числу гнезд высотой 1.

*** Цены указаны в долларах США по состоянию на май-июнь 1990 г.

компьютеров семейства PS/2), при средней пропускной способности 25-30 Мбит/с.

Как обычно, корпорация IBM уделила очень большое внимание программному обеспечению нового семейства компьютеров, поэтому после 15-летнего игнорирования ОС UNIX она была вынуждена, признав свою ошибку, выпустить новую версию 3.0 ОС AIX, созданную специально для семейства компьютеров RS/6000. Разработкой программного обеспечения в среде ОС AIX (и самой ОС AIX) по словам Дейвида МакКензи (David McKenzie), одного из руководителей работ по этой ОС, было занято свыше 2000 программистов (примерно столько же программистов было задействовано на ранних этапах создания программного обеспечения для семейства персональных компьютеров PS/2).

Базовое программное обеспечение семейства RS/6000 специально оптимизировано для работы на компьютерах с архитектурой типа RISC (это коснулось как ОС, так и трансляторов). Полученные результаты превзошли все ожидания - даже специалисты корпорации IBM не рассчитывали, что удастся добиться столь высокого качества объектного кода, генерируемого оптимизирующими трансляторами.

По словам Джона Глайда (John Glyde), одного из руководителей проекта ОС AIX, версия 3.0 ОС AIX является первой разработанной корпорацией IBM ОС, которая соответствует международным стандартам в области мобильных и открытых систем. В частности, это означает:

- соответствие стандартам Posix, SVID и BSD (по исходным текстам);
- включение добавочных 2 млн строк исходных текстов, что обеспечивает возможность работы в режиме реального времени;
- защищенность в соответствии с уровнем C2 так называемой "Оранжевой книги" (*Trusted Computer System Evaluation Criteria*), представляющей собой набор рекомендаций и требований, предъявляемых к аппаратно-программным комплексам, используемым в правительственных учреждениях США;
- прозрачное и автоматическое проведение записи информации на "зеркальный" диск.

Сетевые средства для компьютеров семейства RS/6000 включают в себя поддержку стандартов X.25, TCP/IP и NFS. Все это свидетельствует о попытках корпорации IBM перейти к созданию открытых систем и обеспечению совместимости и преемственности разрабатываемого прикладного программного обеспечения. Реализованы также два интерфейса с пользователем: AIXwindows Environment/6000 и AIX Graphic User Environment/6000 на основе интерфейсов OSF/Motif и NeXTStep соответственно.

К настоящему времени на компьютеры семейства RS/6000 перенесено уже 380 пакетов прикладных программ, разработанных, в основном, для компьютеров RT6150 и семейства PS/2; еще 450 пакетов прикладных программ предполагается перенести к концу третьего квартала 1990 г. Ожидается, что к концу 1990 г. на компьютеры семейства RS/6000 удастся перенести около 1500 пакетов прикладных программ, разработанных 95 различными фирмами (это примерно соответствует номенклатуре пакетов прикладных программ, доступных пользователям рабочих станций, выполненных на базе микропроцессора SPARC фирмы Sun Microsystems).

Корпорация IBM продолжает попытки выйти на рынок рабочих станций. Об этом, в частности, свидетельствует выпуск ею ряда программных продуктов, ориентированных на автоматизацию проектирования, для нового семейства компьютеров RS/6000, работающих под управлением ОС UNIX. Среди этих программных продуктов системы программирования Си++ и Ада, различные кросс-компиляторы, системы обработки текстов, системы планирования и управления проектами и т.д. Все эти программные продукты ориентированы на разработчиков программного обеспечения для встроенных систем, систем реального времени и САПР. Ожидается, что в 1990 г. клиентам будет поставлено около 15 000 компьютеров семейства RS/6000, а в 1991 г. эта величина утроится, что позволит корпорации IBM в 1992-1993 гг. войти в первую четверку производителей рабочих станций.

Таблица 2

Производитель	Компьютер, модель	Результат тестирования (быстродействие в условных единицах)
IBM	RS/6000 Model 540	34.7
IBM	RS/6000 Model 530	28.9
IBM	RS/6000 Model 730	28.9
IBM	RS/6000 Model 930	28.9
IBM	RS/6000 Model 320	22.3
IBM	RS/6000 Model 520	22.3
MIPS Computer	M/2000	17.6
Sun Microsystems	SPARCserver 490	17.6
Solbourne	Series 5/801	16.3
Apollo	DN10010	13.9
Sun Microsystems	Sun SPARCstation 330	11.8
DEC	DECsystem 5400	11.3
DEC	VAX 6000/450	9.2
Sun Microsystems	Sun SPARCstation 1	8.4

Несмотря на столь многообещающие заверения две английские фирмы, занятые разработкой программного обеспечения для компьютеров семейства RS/6000, уже столкнулись с рядом проблем, связанных с функционированием ОС AIX. Тем не менее корпорация IBM продолжает утверждать, что какие-либо проблемы, связанные с программным обеспечением семейства RS/6000, отсутствуют.

Появление компьютеров семейства RS/6000 - по существу новый виток в конкуренции между ведущими фирмами, производящими средства вычислительной техники. Так, например, компьютеры этого семейства конкурируют с компьютерами NeXT, выпускаемыми одноименной фирмой. В ответ на выпуск фирмой IBM семейства RS/6000 фирма DEC выпустила новое семейство рабочих станций с архитектурой типа RISC - DECstation 5000, а фирма Altos - новый компьютер System 5000 на основе микропроцессора Intel 80486 с системной шиной EISA.

ВОКРУГ САПР

Если Вы занимаетесь разработкой многослойных печатных плат (МПП) и имеете возможность использовать компьютер, совместимый с IBM PC, то выбор соответствующей системы автоматизированного проектирования (САПР) Вы уже сделали. Скорее всего, это P-CAD фирмы Personal CAD Systems (США). Если же это не так и Вы применяете другую систему, то, познакомившись с появившейся осенью прошлого года разработкой этой фирмы Master Designer (по привычке называемой отечественными пользователями P-CAD 4.0 или P-CAD 4.5), Вы наверняка измените своей привязанности.

А.Ю. КУЗНЕЦОВ

Несколько советов пользователям P-CAD

САПР для PC, не уступающая САПР для VAX и APPOLO

Чем же привлекателен пакет Master Designer (да и P-CAD вообще) для наших пользователей?

Во-первых, своей универсальностью. Благодаря большому числу драйверов различных периферийных устройств P-CAD можно использовать в системах самой разной конфигурации.

Во-вторых, в зависимости от потребностей пользователей можно довольно широко изменять состав программ, входящих в систему. Например, если Вы занимаетесь только вычерчиванием принципиальных схем или только разводкой, то Вам вполне хватит 2-3 Мбайт на винчестере, в то время как объем полной конфигурации системы может превышать 10 Мбайт.

В-третьих, гибкость системы позволяет конструировать МПП в соответствии с отечественными технологическими требованиями, а ее производительность удовлетворит самых взыскательных пользователей. Вот, например, какие данные приводит фирма в демонстрационной программе к пакету Master Designer:

Программа	Компьютер	Процент выполнения	Время, час
Scicards	VAX	96	1,5
Mentor	Apollo	96	3,5
P-CAD	Compaq 386	96	4,5
Cadnetix	Proprietary	95	2,3
Calma	Apollo	87	Неизвестно

Параметры МПП: 2 слоя, 1284 контакта, 294 цепи, площадь 168 см^2, плотность 0,76 см^2 на одну интегральную микросхему.

И в-четвертых, благодаря процветанию "пиратства" на отечественном рынке программных продуктов информация о правилах работы с системой P-CAD довольно широко распространена в виде переводов фирменной документации пополам с собственными "открытиями", которые можно делать буквально ежедневно. К сожалению, практически полное отсутствие руководств, доступных западным специалистам, или хотя бы публикаций на эту тему в отечественных изданиях превращает в "первооткрывателя" каждого пользователя системы P-CAD. Поэтому мне хочется сделать робкую попытку изменить существующее положение, поделившись наименее, на мой взгляд, распространенными приемами работы с системой P-CAD.

Как бороться с нехваткой памяти?

Иногда, при работе в редакторах PCCAPS, PCCARDS и PCPLACE с командами типа MOV/WIN или ROT/WIN, система выдает сообщения о нехватке памяти или просто отказывается что-либо делать. Каждый пользователь системы P-CAD хотя бы один раз попадал в ситуацию, когда результаты 2- или 3-часовой работы остаются только на экране дисплея и никак не хотят записываться в файл. От этого не застрахованы даже те, кто, зная о рекомендациях фирмы, оставляет как минимум 570 Кбайт свободной памяти и следит за наличием хотя бы 1 Мбайта свободного места на диске, - ведь об этом можно однажды забыть. Можно, конечно, почаще записывать результаты на диск, но радикально решить эту проблему позволит применение платы расширенной памяти и подходящего драйвера. Я воспользовался платой PE505 2.5MB RAM CARD и драйвером ARCEMS.EXE. В руководствах по системе P-CAD рекомендуется использовать расширенную память объемом не менее 1 Мбайт. Проверить правильность работы расширенной памяти можно с помощью команды SYS/STAT в любом из редакторов системы. При работе с P-CAD 3.0 вы увидите, в частности, увеличение макси-

мально возможного числа компонентов (с 500 до 1300), а Master Designer, кроме того, сообщит о наличии LIM-памяти. Если этот путь чем-то Вам не подходит, можно воспользоваться файлами с расширением .CMD, в которых запоминается последовательность работы над базой данных. Самое приятное, что можно даже прервать работу любого из редакторов, одновременно нажав клавиши Ctrl-Alt-Del, - файл *.CMD все равно будет создан и поможет восстановить все Ваши действия. Рассмотрим это на примере редактора PCCAPS. Выйдя из него (любым способом), Вы увидите три файла с именем PCCAPS, два из которых, PCCAPS.~MD и PCCAPS.CMD, можно использовать в качестве командных. Первый из них содержит команды предпоследнего сеанса работы с редактором, а второй - последнего.

Для использования этих файлов следует выполнить три несложные операции:

1) устранить причину, вызвавшую сбой системы P-CAD (очистить память от лишних резидентных программ и драйверов, освободить место на жестком диске);

2) переименовать файл PCCAPS.CMD в файл с расширением .MAC (например, P.MAC);

3) запустить редактор PCCAPS с помощью командной строки

pccaps -r @p.mac

Этим же способом можно бороться и с другими "фатальными" ошибками, например выходом из системы без сохранения результатов или случайным применением какой-либо мощной команды типа DEL/WIN - все зависит от Вашего опыта и фантазии. Очень полезно сохранять командный файл программы размещения компонентов PCPLACE - PCPLACE.CMD, который позволяет быстро повторять процесс размещения.

Программы, без которых можно обойтись, но...

Лучше попробовать их использовать. К таким программам относятся PCDRC.EXE, PCBACK.EXE и PCCOMP.EXE, которых не применяют (по самым различным причинам) многие работающие с системой P-CAD. Эти программы имеют следующее назначение:

- PCDRC.EXE - программа контроля выполнения технологических требований. Очень полезна при проектировании МПП с большим объемом ручной доразводки. Входными файлами для нее являются файл PCDRC.RUL, содержащий технологические требования, и файл базы данных МПП, имеющий расширение .PCB.
- PCBACK.EXE - программа "отката назад" для автоматического внесения изменений в принципиальную схему после размещения компонентов с помощью программы PCPLACE.EXE. Программа PCBACK.EXE генерирует командный файл для редактора PCCAPC.EXE, позволяющий автоматически расставить номера выводов компонентов.
- PCCOMP.EXE - программа, с помощью которой удается сгладить некоторые неудобства при работе с библиотечными элементами: представление информации о компонентах в табличном виде, возможность ее редактировать, а также транслировать информацию из файлов *.SYM в файлы *.PRT значительно упрощают процесс создания библиотек. Мне это упрощение показалось недостаточным, и я написал программу PCLIBR, позволяющую в течение 5 - 10 мин (против 30-40 мин, гарантируемых при работе с системой P-CAD) создавать библиотечные файлы *.SYS и *.PRT для микросхем в планарном и DIP-корпусах с числом выводов от 14 до 48 (см. рекламу ЦПО "МАГИСТР" на с.13 в этом номере "Интеркомпьютера"). Наличие жесткого диалога и проверка практически всех ошибок в процессе ввода информации позволили сберечь нервы и избавиться от лишних затрат времени на поиск ошибок в компонентах.

P-CAD - прекрасная система, а программа трассировки МПП из пакета Master Designer оставляет неизгладимое впечатление. Единственное, что, на мой взгляд, может помешать при выборе P-CAD в качестве САПР, - отсутствие поддержки русских шрифтов, однако это вполне преодолимо: графические файлы из P-CAD можно передать в давно адаптированную к нашим стандартам систему AutoCAD с помощью преобразования формата PLT в формат DXF, применяемого многими пользователями системы P-CAD.

* ФАКТЫ

- Фирма Borland начала поставки новой реализации пакета Turbo C++ в комплекте с отладчиком Turbo Debugger, который располагает средствами отладки прикладных программ, предназначенных для среды Windows. Это первый шаг фирмы Borland к освоению среды Windows, и в скором времени можно ожидать появления компилятора Turbo C++ для Windows.
- Фирма Microsoft начала бесплатную поставку разработчикам программного обеспечения пакета Windows 3.0 для распространения его в комплекте с демонстрационными вариантами программ, созданных этими разработчиками. Таким образом фирма пытается стимулировать разработку и распространение прикладных программ, работающих в мультизадачной среде Windows.
- Стоило фирме IBM на осенней компьютерной выставке в Бостоне (США) продемонстрировать образец 2,5-дюймового жесткого диска емкостью 60 Мбайт, как американское отделение фирмы Toshiba объявило о начале продажи с 1 января 1991 г. своего 2,5-дюймового жесткого диска Mercury 40 емкостью 40 Мбайт.
- Фирма Ashton-Tate разработала версию пакета dBASE IV для ОС UNIX. Эта версия обладает всеми возможностями соответствующей версии для DOS и поддерживает операционные системы SunOS, SCO Xenix, SCO UNIX System V/386, 386/ix. Новая версия популярной базы данных была продемонстрирована на выставке SCO Forum90 в Санта-Круз (Калифорния, США).
- Складывается впечатление, что ведущие фирмы-изготовители компьютеров соревнуются в снижении цен на свои изделия. Так, в начале сентября фирма Dell заявила о том, что ее компьютеры на базе микропроцессоров 80286 и 80386SX подешевеют в среднем на 15%. Например, компьютер Dell System 210 (микропроцессор 80286, тактовая частота 12 МГц, жесткий диск емкостью 40 Мбайт, цветной монитор с адаптером VGA) подешевел на 250 дол. и продается сегодня по 1899 дол. А в конце сентября уже фирма Compaq существенно (в среднем на 20%) снизила цены компьютеров на базе микропроцессора 80386, что в свою очередь было вызвано заявлением фирмы IBM о снижении цен на компьютеры семейства PS/2. Однако рекорд установила фирма Apple: ее компьютер Mac IIci подешевел сразу на 1500 дол.
- Фирма Lotus выбросила на рынок версию 3.1 популярного пакета для обработки электронных таблиц Lotus 1-2-3. Эта версия снабжена некоторыми новыми средствами, в частности новой схемой управления памятью. Тем, кто приобрел версию 3.0 пакета Lotus 1-2-3 после 7 мая нынешнего года, новая версия поставляется бесплатно, а остальным пользователям версии 3.0 - за 35 дол.

ЛИКБЕЗ

Э. РАТЧ

РУКОВОДСТВО для покупателя компьютера 386

Перевод с английского И.В. Емелина

"Интеркомпьютер" получил право на публикацию глав из книги "386 Computer Buyer's Guide and Handbook" американского "технического писателя" (technical writer) Эдвина Ратча (Edvin Rutsch). По-русски название книги звучит несколько витиевато, но оно точно отражает ее содержание, рассчитанное именно на покупателя - не специалиста по компьютерам, а обычного человека, решившего по тем или иным причинам стать владельцем достаточно мощного персонального компьютера. Этим определяются стиль и последовательность изложения. Может быть, искушенным пользователям первые главы покажутся примитивными и неинтересными, однако рекомендуем им запастись терпением и не торопиться с выводами, поскольку в следующих номерах "Интеркомпьютер" познакомит читателей и с нетривиальной информацией, например с тем, как устроены основные блоки компьютера 386, как самостоятельно собрать такой компьютер, как в него превратить обычный XT или AT.

Надеемся, что общение с таким эрудированным и остроумным автором, как Эдвин Ратч, будет для вас приятным!

ВВЕДЕНИЕ

Компьютеры на базе микропроцессора 80386, относящиеся к передовому отряду персональных компьютеров, предоставляют пользователям большие возможности высокопроизводительной обработки данных. В основу этих компьютеров положены новые, все еще меняющиеся технологические решения, что вполне может сбивать с толку широкую публику. Компьютеры на базе микропроцессора 80386 выпускает огромное число фирм, поэтому для многих (а может быть, и для большинства) потребителей задача выбора компьютера может оказаться равносильной научно-исследовательскому проекту солидного размера. Можно потратить дни, а иногда и месяцы, пытаясь осмыслить огромные объемы информации. Кое у кого все это может вызвать приступ головной боли, если не хуже. Потребитель же, заинтересованный в приобретении компьютера, желает знать, как за свои деньги получить наивысшее качество и наибольшую производительность.



Покупка нового компьютера может оказаться радостным событием

Данное руководство должно Вам помочь в выборе наилучшего компьютера в соответствии с Вашими потребностями и возможностями при наименьших затратах времени и усилий на поиски необходимой информации. Оно также может использоваться как справочник, который поможет выжать максимум возможностей из купленного Вами компьютера.

Я начинаю свою книгу с истории развития микрокомпьютеров. Хотя эти сведения и не окажут прямого влияния на Ваше решение при покупке компьютера на базе микропроцессора 80386, тем не менее они помогут узнать, какой была компьютерная индустрия раньше и каковы перспективы ее развития.

Будьте счастливы.

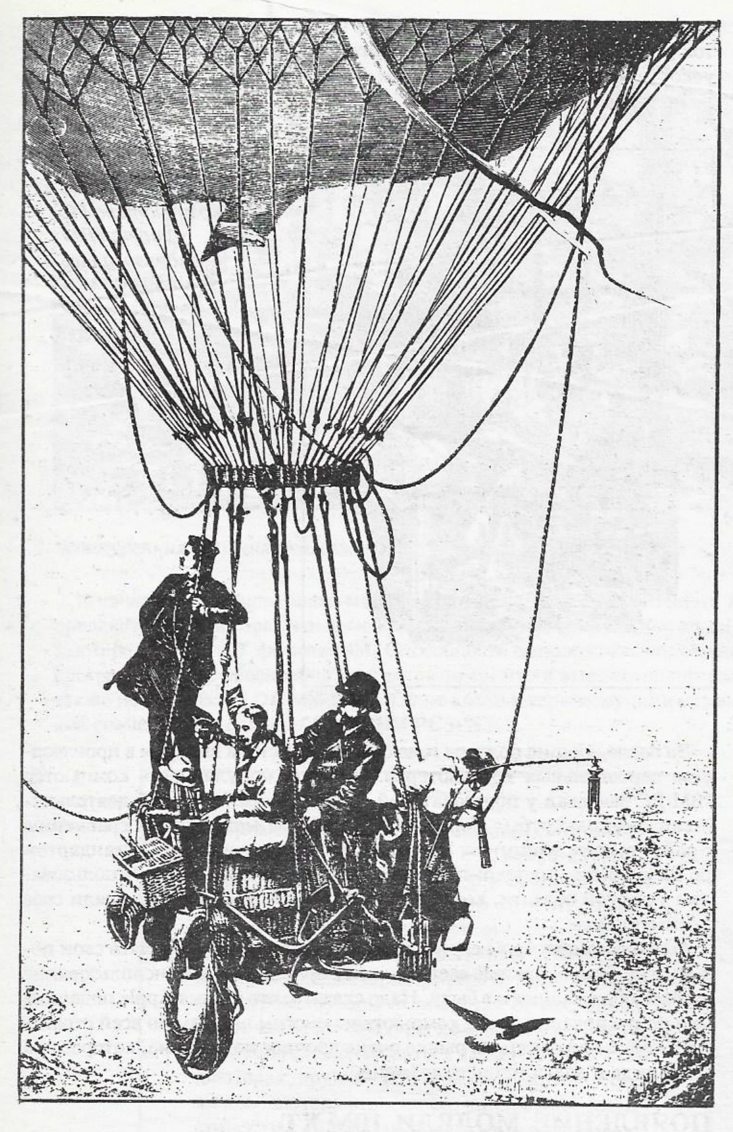
Эдвин Ратч
Сан-Франциско, Калифорния

Глава 1

ИСТОРИЯ РАЗВИТИЯ ПЕРСОНАЛЬНЫХ КОМПЬЮТЕРОВ

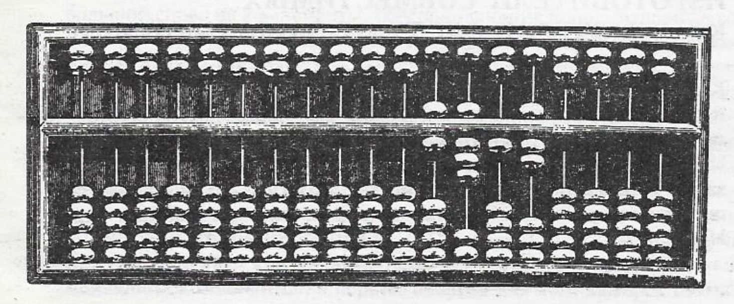
У ИСТОКОВ

"Рождение" персональных компьютеров датируется концом 70-х годов, когда появились первые так называемые микрокомпьютеры. Этот термин был выбран, чтобы подчеркнуть небольшие, "настольные" размеры компьютеров нового типа. До их появления компьютерная индустрия занималась выпуском больших и очень дорогих электронных машин, которые использовались в основном крупными фирмами, достаточно богатыми и мощными, чтобы позволить себе подобную роскошь. Удивительно то, что передовая концепция персонального компьютера была выдвинута не крупной фирмой-законодателем мод, например IBM, а относительно малоизвестными предприимчивыми фирмами: Apple, Morrow, Osborne, Kaypro, North Star (и многими другими).



Обзор прошлого персональных компьютеров может превратиться для Вас в увлекательное приключение

На этапе становления микрокомпьютеров покупателям предлагалось много различных моделей персональных компьютеров, однако только два класса этих машин заслужили одобрение широкой публики: серия компьютеров Apple II и компьютеры с операционной системой CP/M. Их аппаратные части (физические компоненты) существенно различались, и работали они под управлением совершенно разных операционных систем.



Прежним компьютерам операционная система не требовалась

Операционная система представляет собой программу, управляющую работой аппаратных частей компьютера. Существует много типов операционных систем, например CP/M, Apple-DOS, MS-DOS, OS/2, UNIX. Операционная система предоставляет возможность управлять операциями, которые способен выполнить компьютер. Операционная система обычно хранится на гибком диске или на жестком диске.

Одна из основных функций операционной системы состоит в обеспечении возможности выполнения компьютером других программ, в большей степени ориентированных на решение конкретных задач. Они называются прикладными программами, или пакетами программ. Типичными примерами пакетов, предлагаемых покупателям, могут служить те, которые разработаны для решения задач обработки текстов *(word processing)*, управления базами данных *(database management)* и обработки электронных таблиц *(spreadsheet)*. На рынке программного обеспечения имеются тысячи прикладных программ, но все они рассчитаны на выполнение под управлением определенной операционной системы и не способны работать сами по себе. Чтобы программное обеспечение могло выполняться Вашим компьютером, оно должно быть совместимо с имеющейся на нем операционной системой.

ПОЯВЛЕНИЕ КОМПЬЮТЕРОВ СЕРИИ APPLE II

Микрокомпьютеры серии Apple II фирмы Apple быстро завоевали на рынке необычайный успех. Единственная операционная система, которой они снабжались, производила впечатление вполне адекватной своему назначению, однако выпавшему на их долю успеху они были во многом обязаны тому, что попросту оказались первыми жизнеспособными микрокомпьютерами на рынке. Важным фактором, обусловившим этот успех, было и то, что их аппаратная часть во многом была разработана в соответствии с концепцией открытой архитектуры *(open architecture)*. Это значительно усилило полезность и популярность компьютеров серии Apple II.

Открытая архитектура является воплощением очень простой и практичной идеи: электронная машина конструируется с таким расчетом, чтобы пользователь мог ее открыть и расширить возможности путем добавления новых частей или замены уже имеющихся.

Существуют различные уровни открытости архитектуры компьютера. Типичная и наиболее простая модификация компьютера заключается в установке так называемых плат расширения. Чем более открыта архитектура компьютера, тем богаче возможности добавления новых частей или замены старых. Некоторые компьютеры сконструированы таким образом, что все их части могут быть тем или иным путем заменены в целях расширения возможностей этих компьютеров.

Концепция открытой архитектуры позволяет другим фирмам разрабатывать и выпускать отдельные части компьютеров для дополнения или замены, что быстро приводит к интенсивной конкуренции, заставляющей эти фирмы постоянно улучшать свою продукцию. Пользователи компьютеров от этого только выигрывают, поскольку спустя небольшое время на рынке появляется много частей, расширяющих возможности компьютеров.

КОМПЬЮТЕРЫ С ОС CP/M

Операционную систему (ОС) CP/M (Control Program/Monitor - управляющая программа/монитор), разработанную Гарри Килдаллом (G. Kildall), использовали многие фирмы, производящие компьютеры. За определенную плату Килдалл давал им на это право. Такие фирмы, как Morrow, Osborne и Kaypro, остановили свой выбор на ОС CP/M для использования в выпускаемых ими версиях персональных компьютеров.

Хотя компьютеры с этой операционной системой и стали очень популярными на рынке, они получили спрос в основном в мире бизнеса, а не у широкой публики (как компьютеры фирмы Apple). Небольшие предприятия, занимающиеся коммерческой деятельностью, проявляли все возрастающий интерес к относительно недорогим микрокомпьютерам, особенно к компьютерам с ОС CP/M.

ПОЯВЛЕНИЕ КОМПЬЮТЕРОВ IBM PC

На какое-то время компьютеры серии Apple-II и компьютеры с ОС CP/M стали стандартами для индустрии микрокомпьютеров. Они прочно удерживали свои позиции до начала 80-х годов и, вероятно, оставались бы стандартами намного дольше, если бы на рынке персональных компьютеров не появилась фирма IBM.

Когда фирма IBM решила выйти на этот рынок, у нее были большие сомнения в его возможностях. Предполагая, что рынок персональных компьютеров будет в лучшем случае небольшим, фирма решила не тратить слишком много сил на разработку подобного изделия. Она сформировала небольшое подразделение в Бока-Рейтон (Флорида) для выполнения относительно небольшого проекта и предложила его руководителю как можно быстрее выпустить на рынок персональный компьютер. В августе 1981 года фирма IBM (крупнейший в мире изготовитель компьютеров) наконец-то выпустила в свет свой собственный микрокомпьютер.

В своей версии персонального компьютера фирма IBM не использовала ни одной из известных операционных систем - она предложила фирме Microsoft разработать для своего первенца новую операционную систему.

Фирма Microsoft подписала контракт на разработку операционной системы и успешно выполнила его. Фирма IBM внесла в эту операционную систему небольшие изменения и назвала ее PC-DOS (Personal Computer-Disk Operating System - дисковая операционная система для персонального компьютера). В то же время фирма Microsoft сохранила право на продажу разработанной ею операционной системы, которую она назвала MS-DOS (Microsoft-Disk Operating System - дисковая операционная система фирмы Microsoft). Эта операционная система принесла фирме Microsoft большой успех и действительно стала стандартом для операционных систем на рынке персональных компьютеров.

Фирма IBM назвала свой компьютер IBM PC (IBM Personal Computer - персональный компьютер фирмы IBM). На самом деле этот компьютер не стал для рынка чем-то совершенно оригинальным, как это хотела представить фирма IBM. Например, при разработке операционной системы фирма Microsoft использовала многие функции, имеющиеся в ОС CP/M. Кроме того, фирма IBM позаимствовала идею открытой архитектуры у компьютеров серии Apple II.

Хотя фирма IBM славилась тем, что сама производила все необходимое для своих изделий и контролировала все аспекты их применения, при выходе на рынок микрокомпьютеров она изменила этой традиции, воспользовавшись для своего компьютера готовыми (т.е. изготовленными другими фирмами) компонентами. Мотивом для столь серьезного компромисса послужило желание поскорее выпустить на рынок свой персональный компьютер. Благодаря использованию идей и компонентов других фирм процесс разработки компьютера был существенно ускорен, однако при этом патенты и авторские права на многие из них не принадлежали фирме IBM.



При создании компьютера PC фирма IBM воспользовалась компонентами, которые были под рукой

За последующие полтора года фирма IBM стала лидером в производстве персональных компьютеров. Особую популярность компьютер IBM PC завоевал у пользователей, занятых коммерческой деятельностью, поскольку был более перспективной моделью по сравнению с ранее выпускавшимися. Как только этот компьютер стал стандартом de facto среди микрокомпьютеров, предназначенных для использования в данной области, компьютеры с ОС CP/M быстро потеряли свое преимущество.

Однако компьютеры серии Apple II продолжали укреплять свои позиции на рынке, прежде всего в связи с ориентацией на использование в сфере образования и в быту. Надо сказать, что фирма Apple подарила или почти подарила свои компьютеры многим школам по всей стране. Это помогло ей занять на рынке очень прочное положение, которое она сохранила и до настоящего времени.

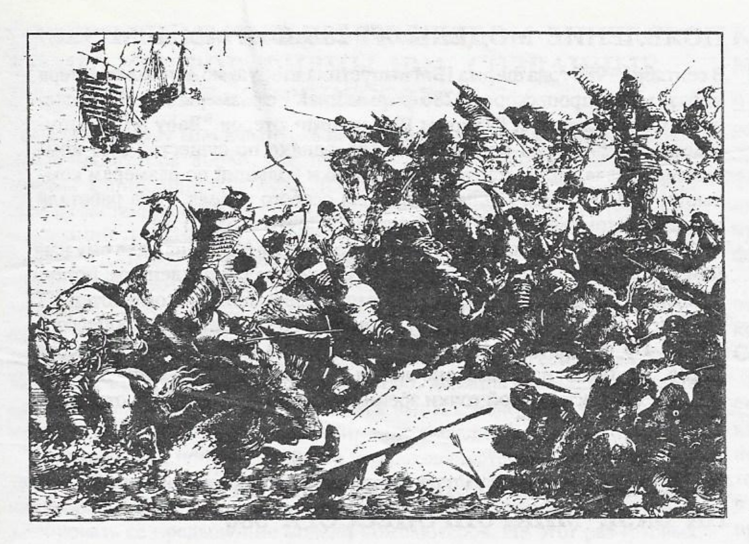
ПОЯВЛЕНИЕ МОДЕЛИ IBM XT

В марте 1983 года фирма IBM выпустила вторую модель микрокомпьютера - IBM XT. Фирма взяла буквы XT из середины слова extra. Акцент на значение этого слова должен был подчеркнуть, что данный микрокомпьютер имеет большие возможности расширения, нежели его предшественник. Если в компьютере IBM PC имелось только пять гнезд для установки дополнительных плат, то в компьютере IBM XT их было восемь. Увеличилась и мощность блока питания: у компьютера IBM PC это были жалкие 63 Вт, а у компьютера IBM XT она уже достигала 135 Вт.

ИЗГОТОВИТЕЛИ СОВМЕСТИМЫХ КОМПЬЮТЕРОВ СЛЕДУЮТ ЗА ФИРМОЙ IBM

Когда гигантская фирма IBM, ставшая законодателем мод, занесла свою пяту над рынком микрокомпьютеров, большинство других изготовителей персональных компьютеров увидели письмена на стене.* Им оставалось либо принять стандарт, предложенный фирмой IBM, либо оказаться лицом к лицу с необходимостью отойти от дел, поскольку большинство выпускаемых пакетов программ было разработано для ОС MS-DOS.

* Имеется в виду библейская история о царе Валтасаре, увидевшем в разгар пира на стене письмена, предвещавшие конец его царствования. - *Прим. перев.*



Конкуренция на рынке компьютеров становится все острее

В качестве ответного шага многие фирмы начали разрабатывать и продавать компьютеры, которые были программно совместимы с компьютерами PC и XT фирмы IBM. Способность выполнять те же самые программы была обеспечена благодаря тому, что в этих компьютерах также использована ОС MS-DOS. Такие компьютеры получили название совместимых *(compatible)* с IBM PC и XT.

Примерами фирм, выпускающих компьютеры, совместимые с PC и XT, могут служить AT&T, Compaq, Epson, Leading Edge, Kaypro и Tandy. Следует отметить, что хотя эти совместимые компьютеры могут выполнять большинство программ, написанных для персональных компьютеров фирмы IBM, но по своему устройству они отличаются от последних. Одни части аппаратуры компьютеров IBM PC и XT могут использоваться в совместимых компьютерах, другие - нет. В зависимости от модели компьютера степень его аппаратной совместимости с IBM PC и XT может меняться.

ПОЯВЛЕНИЕ ИЗГОТОВИТЕЛЕЙ КОМПОНЕНТОВ

За сравнительно короткое время компьютеры IBM XT (а также некоторые модели совместимых компьютеров) завоевали на рынке большой успех. Простота расширения возможностей этих машин определила большой спрос на компоненты, обеспечивающие дополнительные возможности. Американские и азиатские фирмы быстро сориентировались и начали предлагать пользователям таких машин необходимые платы расширения. Эти фирмы начали выпускать дополнительные компоненты, благодаря которым компьютер приобретал новые свойства и улучшались уже существующие (примерами могут служить модемы, дисководы для гибких дисков, мониторы, платы дополнительных портов ввода-вывода, платы памяти и т.д.).

Вскоре после успешного появления на рынке дополнительных компонентов для компьютеров IBM PC и XT и совместимых с ними фирмы-изготовители этих компонентов начали выпускать большие серии заменяющих компонентов, в первую очередь для компьютеров IBM XT. Эти компоненты, производившиеся в основном на Тайване, были достаточно высокого качества, а продавались не более чем за половину цены аналогичных компонентов, поставляемых фирмой IBM.

ПОЯВЛЕНИЕ КЛОНА КОМПЬЮТЕРОВ IBM XT

Так как все дополнительные компоненты для компьютеров IBM XT были почти идентичны и легкодоступны, то возможность сборки компьютеров фирмами-изготовителями компонентов (а также другими фирмами) являлась всего лишь вопросом времени. Собранные ими машины были копиями оригинальных компьютеров IBM XT до такой степени, что даже их внешний вид и габариты были теми же самыми. Семейство подобных компьютеров стали называть клоном компьютеров XT.

Все компоненты компьютеров клона XT были полностью взаимозаменяемыми со всеми (внутренними и внешними) компонентами компьютеров фирмы IBM. Коль скоро компьютеры IBM XT признавались продукцией высокого качества, то же можно было сказать и о компьютерах клона XT, поскольку последние были их точными копиями.

Компьютеры клона XT появились в начале 1985 года и сразу же стали чрезвычайно популярными. Их выпускали в основном азиатские фирмы, которые славились (и продолжают славиться) умением производить дешевую массовую электронную продукцию относительно высокого качества. Некоторые компоненты для компьютеров клона XT изготавливались в США. Обычно события развивались следующим образом: через год или два после того, как американские изготовители выпускали на рынок новый дополнительный компонент для компьютеров фирмы IBM, имеющих открытую архитектуру, азиатские фирмы-изготовители (в первую очередь, тайваньские) начинали выпускать дешевую копию этого компонента.

ЭВОЛЮЦИОННЫЙ ПЕРЕХОД К СЛЕДУЮЩЕМУ ПОКОЛЕНИЮ

Пока фирмы-копировщики были заняты выпуском компьютеров клона XT, фирма IBM не сидела, сложа руки. Ее разработчики интенсивно трудились над созданием следующей модели, которая должна была ознаменовать переход к новому поколению компьютеров.

Слово поколение в данном контексте представляет собой термин, принятый в компьютерной индустрии и имеющий очень специфическое значение по отношению к компьютерам. Когда фирма начинает выпускать новую модель компьютера, это вовсе не означает переход к новому поколению. О новом поколении говорят в том случае, когда фирма выпускает модель компьютера, в которой используется новый тип центрального процессора (ЦП), иногда называемого микропроцессором. В то же время некоторые фирмы-изготовители могут выпускать (и выпускают) различные модели компьютеров на базе одного и того же микропроцессора.

Центральный процессор - это всего лишь один из многих типов электронных микросхем, используемых в компьютерной индустрии. В общем случае микросхема представляет собой маленькое сложное электронное устройство, размещенное на тонкой кремниевой подложке размером не более одноцентовой монеты. Центральный процессор является наиболее важной микросхемой компьютера. Он представляет собой в буквальном смысле его мозг. Основные свойства и возможности компьютера зависят от установленного в нем центрального процессора.

ПОЯВЛЕНИЕ МОДЕЛИ IBM AT

В августе 1984 года фирма IBM "сняла покрывало" со второго поколения своих компьютеров. Она назвала новую модель IBM AT (Advanced Technology - передовая технология). Эта модель была создана на базе центрального процессора, который незадолго до этого был разработан и запущен в производство фирмой Intel (Санта-Клара, Калифорния). Фирма Intel присвоила этой микросхеме номер 80286 (нередко указывается просто 286).

Компьютеры IBM PC и XT были созданы на базе микросхемы 8088 (также фирмы Intel). Разработка микропроцессора 80286, последовавшего за микропроцессором 8088, была качественным скачком в развитии микропроцессоров, так как позволила выпускать более производительные компьютеры, обладавшие большими возможностями. Модель IBM AT была снабжена микропроцессором 80286 и по габаритам превосходила компьютер XT.

Новые компьютеры были с энтузиазмом встречены профессиональными пользователями, занимающимися коммерческой деятельностью, поскольку они уже давно ждали появления машины с большей производительностью. Одним из преимуществ, привлекавших их к компьютерам IBM AT, была совместимость вниз с компьютерами PC и XT. Это означает, что большинство из ранее разработанного программного обеспечения и уже существующие дополнительные аппаратные части могут быть без особых усилий (или почти без усилий) использованы в компьютерах AT. Таким образом, если офисы уже оборудованы микрокомпьютерами фирмы IBM предыдущего поколения, то их можно без проблем дооснастить компьютерами AT. Последние имеют и то преимущество, что обладают не только более высоким быстродействием, нежели их предшественники, но и памятью большего объема.

Фирма IBM сохранила принципы открытой архитектуры и для персонального компьютера AT. Следовательно, этот компьютер также мог быть скопирован ее конкурентами.

ПОЯВЛЕНИЕ КОМПЬЮТЕРОВ, СОВМЕСТИМЫХ С AT

Не прошло и шести месяцев после выпуска в свет оригинальной модели AT фирмы IBM, как фирмы Каурго, Compaq, Sperry и ITT начали производить компьютеры, совместимые с AT. Таким образом, конкуренты снова начали наступать фирме IBM на пятки. Как это уже было с компьютерами предыдущего поколения, компьютеры этих фирм были полностью программно совместимы с IBM AT и лишь частично - аппаратно совместимы.

ПОЯВЛЕНИЕ КЛОНА IBM AT

После появления на рынке компьютеров IBM AT и совместимых с ними компьютеров фирмы-копировщики не заставили себя ждать - они стали производить достаточно качественные, полностью аппаратно совместимые копии оригинального изделия фирмы IBM и продавать их за полцены.

ФИРМЫ-КОПИРОВЩИКИ ВЫПУСКАЮТ "BABY AT"

Начав выпускать собственные версии компьютеров IBM AT, фирмы-копировщики сделали шаг вперед - они представили модифицированную версию компьютера фирмы IBM. Их модель выполняла те же функции, что и IBM AT, но имела габариты компьютера XT. Этого удалось добиться благодаря использованию модифицированного комплекта микросхем, который занимал в корпусе компьютера меньше места.

Оставался открытым вопрос о названии уменьшенной модели компьютера. Строго говоря, он не был представителем клона AT, поскольку у самой фирмы IBM подобной модели не было. Ей давались разные имена: "Baby AT", "Baby 286", "Mini AT" и "Mini 286" (первое из них, по-видимому, стало наиболее популярным).

Вначале спрос на компьютеры "Baby AT" был довольно умеренным, поскольку многие покупатели предпочитали более впечатляющий корпус компьютера AT. Возможно, это объяснялось определенными стереотипами мышления: казалось, что чем больше компьютер, тем выше его производительность, а это, в свою очередь, могло расцениваться как более высокий социальный статус владельца компьютера.

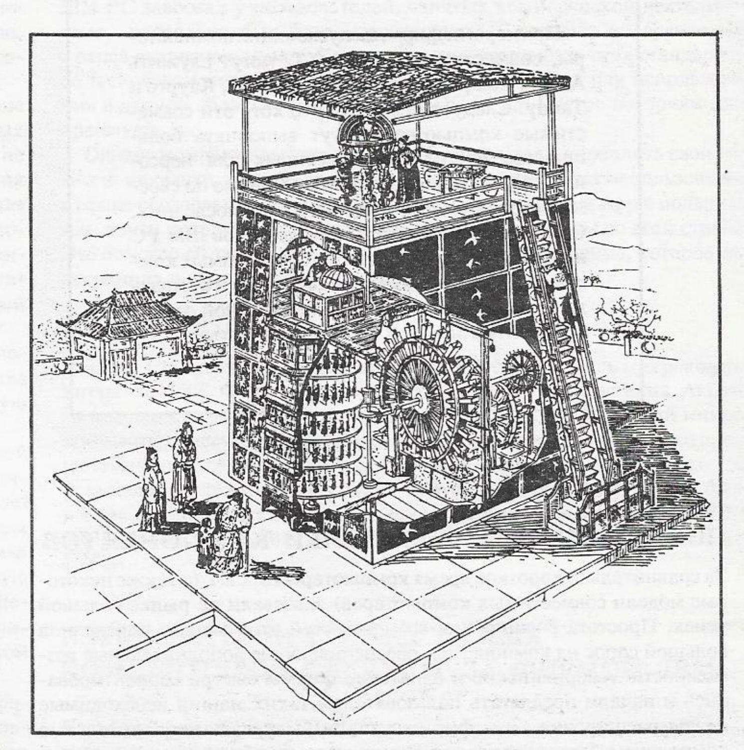
ПОЯВЛЕНИЕ МОДЕЛИ XT 286 ФИРМЫ IBM

В сентябре 1986 года фирма IBM выпустила вторую модель компьютера на базе микропроцессора 80286 фирмы Intel. Ее размеры были теми же, что и у компьютера XT фирмы IBM и компьютеров "Baby AT", выпускавшихся фирмами-копировщиками, однако по существу этот компьютер обладал теми же свойствами, что и больший по размерам компьютер IBM AT. Эта машина не имела особого успеха - она работала медленнее компьютера IBM AT, а стоила почти столько же.

Фирма IBM не назвала этот компьютер "Baby AT". Она дала ему имя IBM XT 286, что, однако, привело к путанице и заблуждениям: потенциальные покупатели думали, что это одновременно и компьютер XT (на базе микропроцессора 8088), и компьютер на базе микропроцессора 80286. На самом деле это был компьютер на базе микропроцессора 80286, но было использовано шасси компьютера XT - отсюда и имя XT 286. С практической точки зрения он вполне может быть отнесен к категории "Baby AT".

РОЖДЕНИЕ ПОКОЛЕНИЯ КОМПЬЮТЕРОВ НА БАЗЕ МИКРОПРОЦЕССОРА 386

Вслед за микропроцессором 80286 фирма Intel выпустила на рынок новый, более производительный микропроцессор 80386, обычно называемый просто 386. Это событие ознаменовало новый шаг вперед в технологии производства центральных процессоров. Предусмотрительные изготовители совместимых компьютеров и фирмы-копировщики выпустили первые модели компьютеров нового поколения (на базе микропроцессора 80386) раньше фирмы IBM, что для многих специалистов оказалось сюрпризом. Эти модели поступили на рынок уже в конце 1986 года.



Один из первых больших компьютеров

С выпуском новых, более производительных персональных компьютеров на базе микропроцессора 80386 сбылись обещания сделать возможности больших компьютеров доступными любому серьезному пользователю. Кроме того, эти компьютеры открыли двери для создания изощренных программ в области искусственного интеллекта *(artificial intelligence)*.

СОВЕРШЕННО НОВЫЕ МОДЕЛИ КОМПЬЮТЕРОВ ФИРМЫ IBM, СОЗДАННЫЕ НА БАЗЕ МИКРОПРОЦЕССОРА 80386

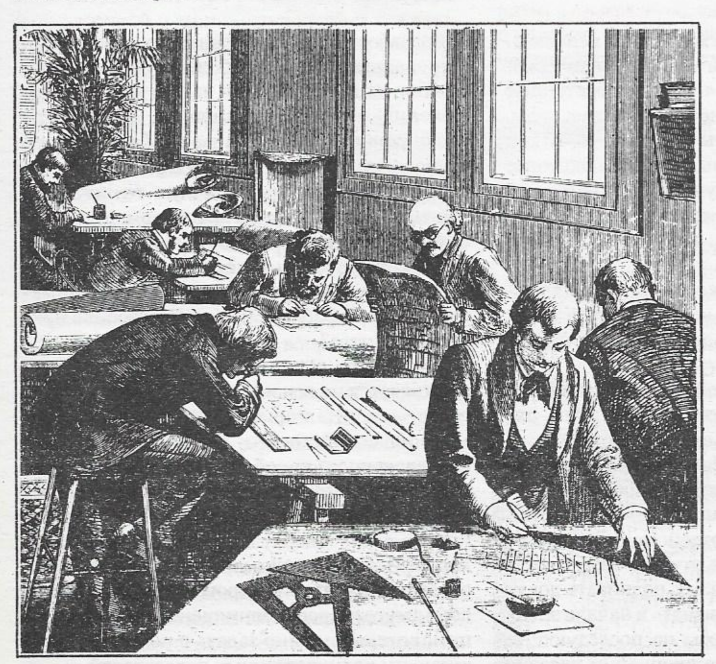
Опередив фирму IBM в выпуске персональных компьютеров на основе микропроцессора 80386, изготовители совместимых компьютеров и фирмы-копировщики нанесли ей чувствительный удар. Однако фирма IBM не заставила себя ждать с ответом. Весной 1987 года она объявила о создании совершенно нового семейства микрокомпьютеров, способного заменить все ее предыдущие модели.

Это семейство определило новый стандарт на аппаратуру микрокомпьютеров, совершенно отличающийся от предыдущего, разработанного самой же фирмой IBM. Новое семейство микрокомпьютеров получило название Personal System/2 (PS/2). Три модели этого семейства, PS/2 Model 30, PS/2 Model 50 и PS/2 Model 60, должны были поступить на рынок немедленно. Осенью ожидался выпуск модели на базе микрокомпьютера 80386 фирмы Intel, а именно модели PS/2 Model 80.

С появлением нового семейства микрокомпьютеров ситуация изменилась. Создавалось впечатление, что фирма IBM пытается исправить некоторые из допущенных ранее ошибок, позволивших другим фирмам копировать ее предыдущие модели компьютеров. На этот раз готовых компонентов она не применяла. В новых компьютерах фирма IBM использовала совершенно новые компоненты, на которые у нее были авторские права. В более мощных компьютерах семейства PS/2 (PS/2 Model 50 и более старшие модели) была использована совершенно иная, улучшенная структура шины, которая как раз и означала попытку введения нового стандарта аппаратной реализации. Таким образом, тысячи ранее разработанных плат расширения уже не могли быть использованы ни в одной из моделей нового семейства компьютеров фирмы IBM.

БУДУЩЕЕ ПОКОЛЕНИЕ КОМПЬЮТЕРОВ НА БАЗЕ МИКРОПРОЦЕССОРА 80486

Фирма Intel продолжала ускоренными темпами разрабатывать новую микросхему центрального процессора. На этот раз речь шла о создании микропроцессора 80486. Компьютеры на его основе были выпущены в 1989 году. Этот микропроцессор также совместим вниз (с предыдущими микропоцессорами), но его производительность в 2-3 раза выше. Пока не ясно, какое влияние окажет выпуск микропроцессора 80486 на компьютерную индустрию, но, по всей видимости, оно будет достаточно велико и приведет к новому скачку в области разработки персональных компьютеров.



Сотрудники фирмы Intel работают над созданием микропроцессора 80486

КОМПЬЮТЕРЫ НЕ НА БАЗЕ МИКРОПРОЦЕССОРОВ ФИРМЫ INTEL

В этой главе сделан обзор истории развития персональных компьютеров, в основном тех, которые были разработаны на базе микропроцессоров фирмы Intel. Необходимо, однако, упомянуть, что есть и другие фирмы, которые разрабатывают и производят микропроцессоры. В индустрии микрокомпьютеров существует еще один стандарт на микропроцессоры. Речь идет о серии микропроцессоров 68000, выпускаемых фирмой Motorola (Феникс, Аризона).

На основе микропроцессоров фирмы Motorola разработано несколько очень популярных моделей компьютеров. Примерами могут служить компьютеры Macintosh фирмы Apple, ST фирмы Atari и Amiga фирмы Commodore.

Компьютеры на основе микропроцессоров фирмы Motorola предоставляют пользователям богатые графические возможности, в то время как компьютеры на базе микропроцессоров фирмы Intel используются преимущественно в деловых приложениях, например для обработки текстов, ведения баз данных и обработки электронных таблиц. Однако в последнее время компьютеры на базе микропроцессоров фирмы Intel проникают и в области графической обработки.

Изготовители компьютеров на базе микропроцессоров фирмы Motorola разработали собственные независимые операционные системы, что сделало их компьютеры совершенно несовместимыми с компьютерами других фирм, даже если в последних использован тот же самый микропроцессор фирмы Motorola. Результатом является относительно бедный ассортимент программного обеспечения для этих компьютеров. Их владельцы не могут воспользоваться ни одним из тысяч пакетов программ, пригодных для компьютеров на базе микропроцессоров фирмы Intel.

ПОЯВЛЕНИЕ КЛОНОВ КОМПЬЮТЕРОВ СЕМЕЙСТВА PS/2 ФИРМЫ IBM И СОВМЕСТИМЫХ С НИМИ КОМПЬЮТЕРОВ

В апреле 1988 года две фирмы, Tandy и Dell Computers, начали выпуск компьютеров, совместимых с моделями семейства PS/2 фирмы IBM, выполненными на базе микропроцессоров 80286 и 80386. Многие другие американские и азиатские изготовители объявили о своем намерении выйти на рынок с аналогичными изделиями в конце 1988 года. Тем самым было положено начало новому стандарту аппаратной реализации, который, по мнению многих экспертов, должен за несколько лет стать доминирующим. Однако фирма IBM защитила свои новые компьютеры патентами, чтобы изготовителям совместимых компьютеров и фирмам-копировщикам было не так легко украсть успех у компьютеров семейства PS/2, как им это удалось сделать с компьютерами XT и AT.

БУДУЩЕЕ МИКРОКОМПЬЮТЕРОВ

Последние маневры фирмы IBM на рынке микрокомпьютеров вызвали ряд вопросов. Сможет ли фирма IBM укрепить свои позиции благодаря новому стандарту аппаратной реализации? Создадут ли шустрые конкуренты клоны и смогут ли отобрать у фирмы IBM если не весь, то хотя бы часть рынка, как это уже было сделано ими раньше? Насколько будет велик спрос на эти компьютеры? Не совершите ли Вы непоправимой ошибки, если закажете компьютер из числа выполненных в соответствии с прежним стандартом?

Эти и многие другие вопросы о будущем микрокомпьютеров в настоящее время не сходят с уст экспертов, пользователей и в особенности покупателей компьютеров. Многие из них будут рассмотрены в следующих главах.

Эдвину Ратчу 34 года. Он живет в Сан-Франциско (Калифорния) с родителями, выходцами из Германии. Много путешествует по миру. Именно охотой к перемене мест можно объяснить тот факт, что Ратч успел поучиться в четырех американских университетах, а также университетах ФРГ и Индонезии.

Кроме той книги, которую мы сегодня предлагаем вниманию читателей, Ратч написал еще две: для пользователей компьютеров XT и AT.

В июне нынешнего года Эдвин Ратч был гостем московского Международного компьютерного форума.

ЛИКБЕЗ



М.А. БОГДАНОВ

Криптография и компьютеры.
Что их связывает?
Что дает этот альянс обеим сторонам?
Насколько он прочен и каково его будущее?
Какие возможности открывает этот альянс
для разработчиков компьютеров
и пользователей?
Что получат пользователи?
Какие при этом возникают проблемы?

КРИПТОГРАФИЯ и КОМПЬЮТЕРЫ

Сегодня компьютерная обработка информации ("безбумажная технология") проникает во все сферы, связанные с хозяйственной деятельностью предприятий и организаций, а также частной жизнью людей. Обрабатываемая компьютерами информация всегда представляла определенный интерес для охотников за коммерческими и промышленными секретами, для лиц, занимающихся мошенничеством и шантажом, да и просто для любопытных. В современных условиях перехода нашей страны к рыночной экономике и возникающей в связи с этим конкуренцией такая информация становится еще более притягательной, а это в свою очередь повышает роль защиты информации, подвергаемой компьютерной обработке.

Компьютеризация и криптозащита информации

Необходимость обеспечения защиты информации при использовании "безбумажной технологии" не вызывает сомнений. Однако особенности электронной формы представления информации обусловливают поиск и разработку качественно новых решений проблем ее защиты, нежели сложившиеся при традиционной "бумажной технологии". Действительно, для защиты информации, представленной в электронной форме, уже нельзя использовать ориентированные на наличие физического носителя средства и методы, изобретенные и применяемые человечеством многие годы. Охрана и сейфы, пакеты и папки, печати и пломбы, штампы и личные подписи, водяные знаки и т.д. в наступившую эру "безбумажной технологии", как правило, малоэффективны, а в отдельных случаях и принципиально неприменимы.

Например, как реально обеспечить надежную защиту многокилометровых проводных каналов связи, по которым передается информация?

А как быть с радиорелейными линиями, радиоканалами и спутниковыми каналами, охранять которые в принципе невозможно?

Как обеспечить конфиденциальность информации и доверие к ней пользователей, если заведомо известно, что во время обработки может осуществляться воздействие нарушителя (злоумышленника) на эту информацию?

Каким образом можно так подписать документ и поставить на нем печать в начале электронной обработки, чтобы на последующих этапах обработки и по ее окончании указанные подписи и печати удостоверяли подлинность документа в общепринятом юридическом смысле? (Последняя проблема - это проблема так называемой "цифровой подписи".)

Разработчики современных вычислительных средств и систем все больше усилий затрачивают на поиск ответов на эти вопросы, без решения которых немыслимо дальнейшее расширение сферы применения безбумажной технологии. За последние годы было найдено множество достаточно удачных технических решений в области защиты информации. Особый интерес представляют решения, основанные на применении методов криптографии.

Чем же вызвано такое повышенное внимание к криптографии? Что в ней особенно привлекательного?

Во-первых, если защита информации осуществляется на основе эффективного криптографического алгоритма, прошедшего криптографический анализ, т.е. предварительно исследованного какой-либо экспертной организацией, и при этом выполняется ряд обязательных условий, то специалисты по защите информации говорят о *гарантированной защите.* Под этим понимается возможность построения достаточно строгого доказательства достигнутого уровня защищенности информации, который можно задать с помощью совокупности количественных показателей.

Во-вторых, применение криптографических методов позволяет обеспечить такой высокий уровень защищенности информации, который нельзя получить с помощью

некриптографических методов и средств защиты.

И наконец, в-третьих, криптография способна обеспечить эффективное решение проблемы защиты информации в тех случаях и при таких предположениях о возможностях нарушителя, когда некриптографические методы либо вовсе неприменимы, либо баснословно дороги.

Следует отметить, что применение методов криптографии для защиты информации, или, другими словами, криптографическая защита информации обеспечивает достижение следующих двух (или каждой в отдельности) целей:

- сохранение конфиденциальности, т.е. обеспечение секретности информации до момента снятия с нее криптографической защиты;
- возможность обнаружения (с заданной вероятностью) преднамеренных искажений информации, внесенных нарушителем до момента снятия с нее криптографической защиты.

При этом необходимым условием достижения указанных целей является обеспечение секретности и целостности используемых криптографических ключей. Следует подчеркнуть, что объем памяти, занимаемый криптографическим ключом, значительно меньше суммарного объема защищаемой с его помощью информации - он составляет несколько десятков байт.

Демократизация криптографии

Едва ли нужно говорить о том, что криптография - "особая" научная дисциплина. Достижения в этой области до последнего времени были закрыты для широкого круга специалистов, занимающихся проблемами электронной обработки и обмена информацией, да и само разрешение на применение криптографии в военной и дипломатической связи являлось исключительной прерогативой государства. Однако, в 1970-х годах ситуация стала резко меняться.

С началом информационного бума и проникновением компьютеров и сетей компьютеров в хозяйственную деятельность и личную жизнь людей все более широкие слои общества стали осознавать необходимость применения методов криптографии для решения проблем обеспечения защиты информации. Это обстоятельство подтверждается резким увеличением за последнее десятилетие числа публикаций, посвященных вопросам криптографии, как в научных, так и в научно-популярных журналах, что повысило осведомленность людей в этой области. Из последних публикаций в нашей стране, по мнению автора наиболее полные сведения приведены в специальных выпусках журналов ТИИЭР (N 5 за 1988 г.) и "Зарубежная радиоэлектроника" (N 12 за 1989 г.).

Для объединения усилий специалистов, работающих в этой области, почти десять лет назад в США создана Международная ассоциация криптографических исследований (International Association of Cryptographic Research - IACR). Результаты демократизации криптографии не заставили себя ждать. Появился целый ряд принципиально новых оригинальных идей. Заложены основы нового направления в криптографии, которое в отличие от классической криптографии получило название *криптографии с открытым ключом*. Не вдаваясь в детали, здесь хотелось бы отметить, что именно криптография с открытым ключом позволит решить проблему "цифровой подписи" - электронного аналога личной подписи и проблемы рассылки криптографических ключей, что, видимо, произведет коренную ломку в сложившихся представлениях о способах создания безопасных компьютерных сетей с сотнями тысяч абонентов.

Крупным практическим шагом в направлении демократизации криптографии за рубежом явилось появление общедоступных (несекретных) криптографических алгоритмов, предназначенных для защиты информации, подвергаемой компьютерной обработке. Наиболее удачный криптографический алгоритм, разработанный фирмой IBM в рамках классической криптографии, уже несколько лет назад принят в США в качестве государственного стандарта (Data Encryption Standard - DES) для использования в коммерческих системах, что положило начало разработке целого ряда стандартов США в области криптографии.

В рамках криптографии с открытым ключом наилучшим следует признать криптографический алгоритм RSA, названный по фамилиям его разработчиков (R. Rivest, A. Shamir, L. Adleman).

Средства, обеспечивающие криптографическую защиту информации, можно реализовать традиционным аппаратным способом (в виде специализированных электронных устройств), программным способом, а также программно-аппаратным способом. При использовании программно-аппаратного способа аппаратно (в виде внешних устройств или устройств, встраиваемых в компьютер) реализуются в первую очередь функции, связанные с хранением применяемых криптографических ключей, и сам криптографический алгоритм, а программно, как правило, - интерфейс с пользователем и сервисные функции по управлению и контролю функционирования.

ФАКТЫ

Фирма Digital Research воспользовалась некоторым замешательством, возникшим при подготовке очередной версии MS-DOS, и выпустила новую версию собственной операционной системы DR DOS 5.0. Эта версия полностью совместима с операционной системой MS-DOS, и под ее управлением можно запускать все прикладные программы, разработанные для MS-DOS. Официальный представитель фирмы Digital Research заявил, что в версии VS-DOS, которую фирма Microsoft обещает выпустить лишь в начале 1991 г., будут скопированы многие возможности, реализованные в DR DOS 5.0, например средства управления расширенной памятью.

Управление по защите информации США ежеквартально выпускает несекретный каталог изделий и услуг для защиты информации (Information Systems Security Products and Services Catalogue). В него включено шесть списков изделий и услуг, ранее издававшихся различными подразделениями Управления национальной безопасности США, в том числе:

- список криптографических изделий, одобренных к применению (Endorsed Cryptographic Products List);
- список изделий на основе алгоритма DES, одобренных к применению (NSA Endorsed DES Products List);
- список компаний, предоставляющих защищенные каналы связи (Protected Service List);
- список изделий, прошедших оценку (Evaluated Products List).

Кроме перечисленных в состав каталога входит ряд других списков.

С помощью этого каталога любая правительственная организация или частная фирма может выбрать изделия (услуги), необходимые для удовлетворения стандартных требований по защите информации.

К настоящему времени рядом фирм выпущены многочисленные типы специализированных микросхем высокой степени интеграции для аппаратной реализации ряда криптографических алгоритмов, в том числе и в соответствии со стандартом DES.

Есть примеры довольно удачной программной реализации указанного криптографического алгоритма.

Эти средства позволяют удовлетворять всевозрастающие потребности в криптографической защите не только фирм США, Канады, но и ряда стран Западной Европы.

Аналогичные работы по стандартизации криптографических алгоритмов и созданию специализированных микросхем для их реализации, хоть и с некоторым опозданием, проводятся также в нашей стране. Получены обнадеживающие результаты как в области стандартизации криптографических алгоритмов, так и в области их практической реализации, что позволяет надеяться на удовлетворение в ближайшем будущем потребностей народного хозяйства нашей страны в криптографической защите информации.

Возникло и развивается сближение криптографии и компьютеров, явно полезное обеим сторонам. И уже сейчас можно с полной уверенностью утверждать, что дальнейшая компьютеризация общества немыслима без этого альянса.

Основные области применения

Области применения криптографии при внедрении безбумажной технологии настолько разнообразны, что этот вопрос может стать предметом отдельного обсуждения. Здесь кратко упомянем лишь основные из них.

1. Криптографическая защита информации, передаваемой по каналам связи. Это ос-

новная область применения криптографии. Существует множество методов криптографической защиты передаваемой по каналам связи информации, что обусловлено многочисленными вариантами требований, предъявляемых пользователями к защите информации, различием характеристик потоков информации, типов каналов связи, форматов сообщений и способов доведения информации до адресата (например, циркулярно, избирательно и т.д.).

Актуальность этой области применения криптографии в ближайшем будущем резко возрастет ввиду наметившейся в нашей стране тенденции массового подключения компьютеров к сетям общего пользования, в том числе к городским и междугородным сетям.

2. Криптографическая защита данных, записанных на встроенные в компьютеры или съемные внешние носители. Это позволяет обеспечить конфиденциальность информации как при утере или краже съемного носителя, так и при попытках несанкционированного чтения данных, а также эффективный контроль целостности данных и программ. Последнее является действенным средством, предотвращающим проникновение "компьютерных вирусов", которые сегодня стали поистине бедствием для персональных компьютеров.

3. Построение парольных систем аутентификации (опознания) пользователей отдельных компьютеров и абонентов компьютерных сетей.

Криптографические методы позволяют хранить таблицу паролей в зашифрованном виде. Более того, с помощью методов криптографии можно построить такую процедуру аутентификации пользователей, для которой вообще не требуется таблица паролей. Достаточно знать лишь криптографический ключ, так как в этом случае требуемый пароль формируется с помощью криптографического алгоритма в процессе аутентификации.

4. Программная реализация генератора массива псевдослучайных чисел гарантированного качества. Такой генератор необходим для решения вспомогательных задач защиты информации, например для создания таблиц паролей, криптографических ключей и т.д. Идея этой реализации состоит в том, что любой массив, зашифрованный с помощью криптографического алгоритма, по существу представляет собой массив псевдослучайных чисел, доказательство случайности которых не требует проведения громоздких и трудоемких статистических проверок.

Основные этапы реализации криптографической защиты

Задачи, с решением которых сталкиваются разработчики компьютеров и компьютерных систем, применяя методы криптографии, можно решать в несколько этапов:

1. *Выбор средств криптографической защиты.* Следует подчеркнуть необходимость создания справочников и каталогов, содержащих перечни существующих и разрабатываемых средств криптографической защиты, прошедших аттестацию в каких-либо компетентных организациях (т.е. получивших соответствующий сертификат), их назначение и основные технические характеристики.

2. *Выбор места подключения аппаратных средств криптографической защиты к компьютеру или компьютерной системе (с учетом электрического сопряжения).*

3. *Логическое сопряжение*, т.е. интерфейс процедуры реализации криптографической защиты с процессом обработки информации. В рамках решения этой задачи разрабатывается протокол организации криптографической защиты, содержащий перечень всех тех правил и характеристик, которые являются стандартными для такого рода документов: формат блока данных (или модуля программы), подвергаемого защите за одно обращение к средствам криптографической защиты, критерии проверки подлинности информации и правила выбора соответствующего криптографического ключа. На этом этапе формулируются также требования и ограничения на последующий после криптографической защиты процесс обработки информации.

4. *Построение ключевой системы, соответствующей принятому в данном случае методу реализации криптографической защиты.* Задача построения ключевой системы состоит в описании используемых типов ключей, способов их формирования и распределения между абонентами компьютерных систем и пользователями компьютеров, правил использования и уничтожения ключей.

5. *Анализ выполнения технических требований по обеспечению криптографической защиты.* В рамках этого заключительного этапа реализации криптографической защиты формулируются условия, обеспечение которых обязательно для правильного применения криптографических методов, дается перечень организационно-технических мероприятий. Результатом анализа является заключение, которое представляет собой юридический документ, передаваемый в экспертную организацию. При отрицательном результате экспертизы проводится доработка средств защиты информации в соответствии с полученными рекомендациями экспертов. Экспертиза - это важная составная часть цикла работ по применению методов криптографии.

Желательно как можно скорее решить вопрос о создании ряда коммерческих экспертных организаций, что позволит расширить сферу применения криптографических методов защиты информации.

Одновременно надо решить вопросы, связанные с разработкой, изготовлением и рассылкой потребителям криптографических ключей.

В заключение следует подчеркнуть, что принимая решение об использовании методов и средств криптографической защиты информации, нужно быть уверенным, что оно продиктовано не данью моде и не увлечением экзотическим средством защиты информации, а является результатом серьезного технико-экономического обоснования.

И, наконец, более полное представление о поднятых в данной статье проблемах требует детального освещения целого ряда вопросов. Вот далеко не полный их перечень:

- выбор криптографического алгоритма, ориентированного на компьютерную реализацию;
- способы реализации криптографического алгоритма;
- протокол организации криптографической защиты;
- пользовательский интерфейс со средствами криптографической защиты;
- вопросы реализации ключевой системы;
- порядок проведения экспертизы системы криптографической защиты.

СЛОВАРЬ

attacker (*нарушитель, противник*) - противодействующая сторона, преднамеренные действия которой нарушают безопасность информации в рассматриваемой компьютерной системе.

authentification (*аутентификация*) - установление подлинности информации (сообщений, файлов, данных), идентификаторов пользователей, терминалов, сетей и т.д.

central keying authority - центр распределения (распространения, управления) ключей, ЦРК.

cryptoanalysis - дешифрование, криптоанализ.

cryptography (*криптография*) - наука о методах синтеза и анализа криптографических систем, криптографических алгоритмов преобразования знаковой информации к виду, в котором ознакомление с ее содержанием и необнаруживаемое искажение (подделка) невозможны без знания отдельных элементов этого преобразования, называемых криптографическим ключом.

cryptosecurity (*криптографическая защита*) - защита информации путем осуществления ее криптографического преобразования.

cryptosystem - криптографическая система.

decryption - расшифрование.

digital signature (*аутентификатор, цифровая подпись*) - дополнительная информация, предоставляемая источником для обеспечения аутентификации.

encryption - (за)шифрование.

endorsment - аттестация.

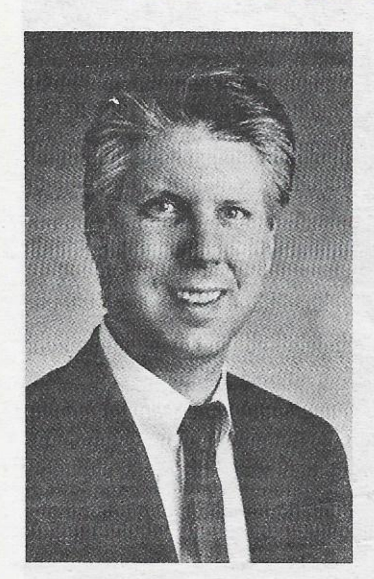
forgery (*подделка информации*) - несанкционированное искажение информации.

key distribution - распределение (распространение, доставка) ключей.

key management (*ключевая система*) - совокупность криптографических ключей и правил обращения с ними при обеспечении криптографической защиты информации.

security - безопасность, защита.

ГОВОРЯТ УЧАСТНИКИ МЕЖДУНАРОДНОГО КОМПЬЮТЕРНОГО ФОРУМА



Фирма Zeos International (США) должна быть хорошо знакома тем советским специалистам, которые хотя бы иногда перелистывают наиболее популярные зарубежные компьютерные журналы. На их страницах фирма или представлена рекламой, или упоминается среди ведущих изготовителей IBM-совместимых компьютеров, высокое качество, надежность, производительность и конкурентоспособность которых известны во всем мире.

Продукция фирмы удостоена наград многих компьютерных журналов.

Сейчас в США особенно известна модель Zeos 386SX-16.

На вопросы нашего корреспондента отвечает президент фирмы Zeos International Грег Херрик (Greg Herrick).

- Какие соображения привели Вас на этот форум?

Грег Херрик: Мы здесь для того, чтобы представить наши продукты на советском рынке. Мы, безусловно, заинтересованы в выходе на советский рынок, но наряду с этим нам еще очень интересно просто узнать о нем побольше. Интересно обменяться идеями, познакомиться с представителями различных предприятий и организаций в Советском Союзе, узнать их возможности и потребности.

- До приезда на форум Вы, наверное, имели какую-то информацию о советском рынке. Сильно ли она отличается от той, которой Вы располагаете теперь?

Г.Х.: Как оказалось, советские пользователи точно так же, как и американские, заинтересованы в том, чтобы было "больше, лучше, быстрее". В частности, имеется большая потребность в промышленных системах, системах для управления производством и научных исследований. В этом плане советский рынок ничем не отличается от американского, и это как раз то, чего я и ожидал. Однако кроме этого у советских пользователей огромный интерес к программным продуктам, хотя это немного странно, так как советские программисты - это, как нам кажется, один из самых мощных ресурсов, имеющихся в Советском Союзе.

Мы видим для себя еще несколько перспективных возможностей, которых не видели раньше. Мы могли бы, например, предложить компьютерные системы, существенно более дешевые, чем применяемые сейчас, и для них в Советском Союзе существует очень широкий рынок потребителей. Вернувшись в США, я буду подробно обсуждать все эти идеи с коллегами. Очень рассчитываю на сотрудничество с Тестовой лабораторией МКК, где мы сможет проверить наши концепции, касающиеся дешевых систем.

- Поскольку сейчас мы тесно сотрудничаем с Международным компьютерным клубом, то будем публиковать результаты тестирования в нашем журнале, где для этого открыта специальная рубрика. Теперь последний вопрос: "Каковы Ваши планы, цели и задачи на советском рынке?"

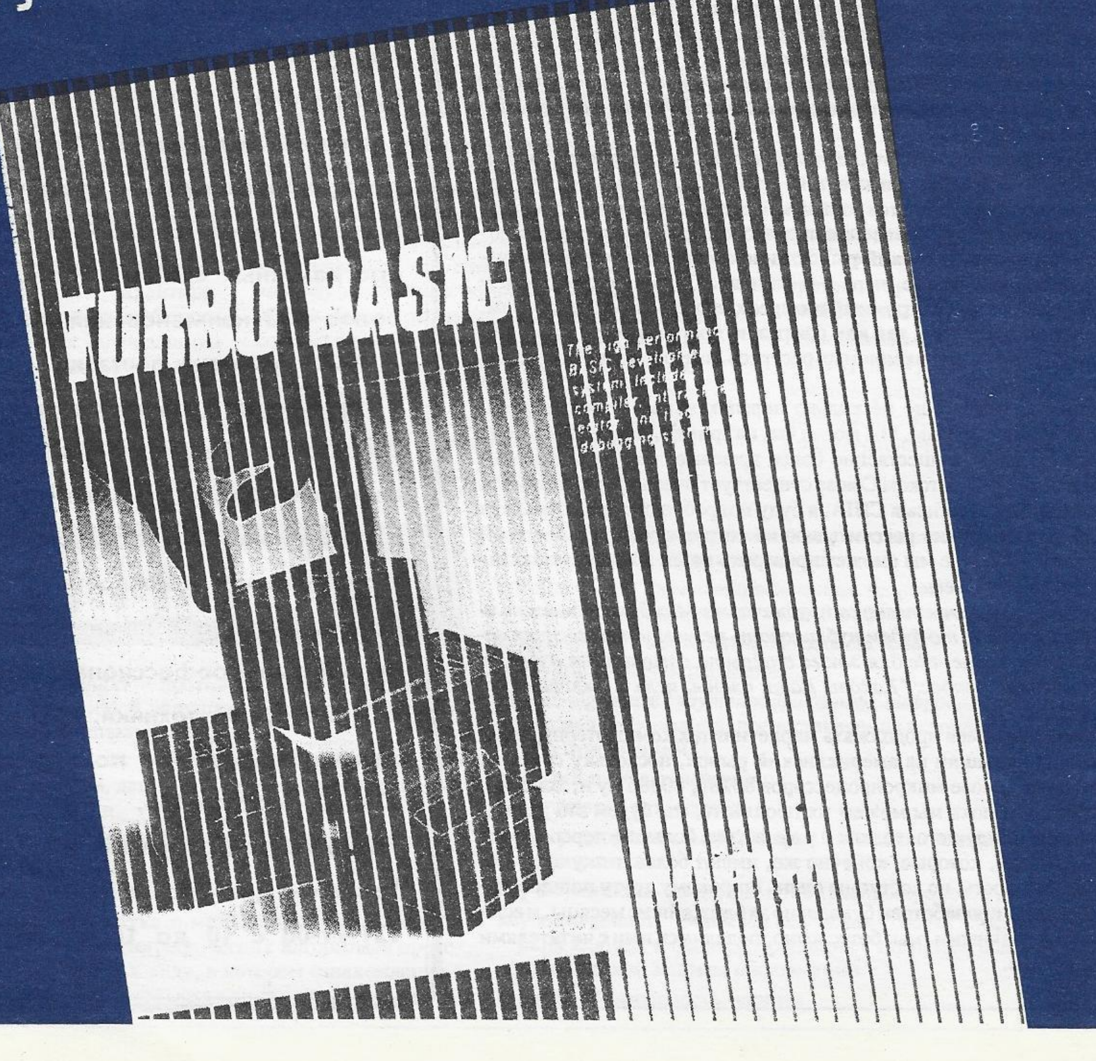
Г.Х.: Мы собираемся продолжать маркетинг тех компьютерных систем, с которыми вышли на американский рынок, поскольку спрос на компьютеры на основе микропроцессоров 80286, 80386 ну и, конечно, 80486 огромен, и пока мы можем их продавать, мы будем это делать. Что же касается будущего, то, как я уже сказал, большие перспективы у дешевых систем, которые, конечно же, имеют более низкую вычислительную мощность, но доступны очень широкому кругу пользователей. Эти идеи мы проработаем буквально за ближайшие месяцы, и если они окажутся удачными, мы, безусловно, поделимся ими с читателями Вашего журнала.

БИБЛИОТЕЧКА ИНТЕРКОМПЬЮТЕРА

Б. А. Ашкинази

TURBO BASIC

Руководство для пользователей



Окончание (начало см. в двух предыдущих номерах).

2.7. ПОДПРОГРАММЫ, ПРОЦЕДУРЫ, ФУНКЦИИ

Подпрограммы

В терминологии языка Turbo Basic подпрограмма - это выполняющая конкретную функцию группа операторов, к которой можно обратиться из разных мест программы.

Такая группа представляет собой последовательность операторов, заканчивающуюся оператором RETURN.

Формат:

RETURN

Переход к подпрограмме осуществляется с помощью оператора GOSUB.

Формат:

GOSUB label

Здесь label - метка первого оператора подпрограммы. К подпрограмме нельзя переходить с помощью оператора GOTO, и нельзя выходить из подпрограммы иначе, как с помощью оператора RETURN.

Процедуры

Процедуры в языке Turbo Basic отличаются от подпрограмм тем, что допускают использование параметров. Как и подпрограмма, процедура - это группа операторов, но первым должен быть оператор SUB, а последним - оператор END SUB.

Формат:

```
SUB name(param1,param2,...)
END SUB
```

Здесь:
name - имя процедуры;
paramN - параметр (имя переменной или массива).

К процедуре обращаются с помощью оператора CALL.

Формат:

CALL name(param1,param2,...)

Внутри процедуры можно использовать локальные (LOCAL), статические (STATIC) или глобальные (SHARED) переменные и массивы.

Указание типа памяти

Операторы SHARED, LOCAL и STATIC позволяют указывать тип памяти для используемых в процедуре переменных и массивов.

Формат:

```
SHARED var1,var2,...,array5
LOCAL var1,var2,...,arrya5
STATIC var1,var2,...,array5
```

Обьекты с типом памяти LOCAL и STATIC локальны для процедуры, причем при выходе из процедуры значения обьектов с типом памяти LOCAL уничтожаются, а значения обьектов с типом памяти STATIC сохраняются. Обьявление обьекта как имеющего тип памяти SHARED говорит о том, что он содержится в программе, вызвавшей процедуру.

Следует заметить, что в подпрограмме все обьекты имеют тип памяти SHARED (при этом их не нужно обьявлять).

Рекурсия

Процедуры могут быть рекурсивными, подпрограммы - не могут.

Директива $STACK

Память для переменных процедур выделяется в стеке. При большом числе переменных обьема стека, выделенного по умолчанию (1024 байт), может не хватить. В этом случае его можно изменить с помощью директивы $STACK.

Формат:

$STACK size

Функции

Кроме подпрограмм и процедур в языке Turbo Basic имеются также функции. Они очень похожи на процедуры, но могут возвращать значения.

Формат:

```
DEF FNname(param1,param2,...)
statements
DEF FNname = xxxxx
END DEF
```

Первый оператор определяет имя (name) и параметры функции. Далее идут собственно операторы функции. Предпоследний оператор присваивает функции значение, последний оператор заканчивает ее определение.

По умолчанию все обьекты внутри функции глобальные, но с помощью операторов LOCAL и STATIC можно описать локальные и статические обьекты.

Выход из функции можно осуществить с помощью оператора EXIT DEF.

Как и процедуры, функции могут быть рекурсивными.

Стандартные функции

Как и в других языках программирования, в языке Turbo Basic есть набор стандартных (ранее предопределенных) функций, список которых можно получить с помощью подсистемы оперативной помощи:

ABS(X) - абсолютное значение;
ATN(X) - арктангенс;
COS(X) - косинус;
EXP(X) - степень (натуральная);
EXP2(X) - степень (двоичная);
EXP10(X) - степень (десятичная);
FIX(X) - целая часть числа;
INT(X) - целая часть числа с округлением (например, INT(-3.4) = -4);
LOG(X) логарифм;
MOD(X) остаток от деления;
SIN(X) - синус;
SGN(x) - знак числа (например, SGN(-3) = -1);
TAN(X) - тангенс.

2.8. СЛУЧАЙНЫЕ ЧИСЛА

Для работы со случайными числами в языке Turbo Basic используются функция RND и оператор RANDOMIZE.

Функция RND

В зависимости от значения аргумента X функция выполняет следующие действия:

X > 0 - возвращает случайное число в диапазоне от 0 до 1;
X < 0 - инициализирует серию случайных чисел.

Формат:

RND(X)

Оператор RANDOMIZE

В зависимости от значения аргумента инициализирует серию случайных чисел.

Формат:

RANDOMIZE number

2.9. СТРОКИ, ДАННЫЕ, МАССИВЫ

Строки символов

В языке Turbo Basic основным средством хранения и обработки нецифровой информации являются строки символов. В именах переменных, предназначенных для хранения строк, последним символом должен быть символ $. Длина строки не может превышать 32767 символов.

Для сравнения строк применяют обычные операторы сравнения (, = и т.д.). Для объединения (конкатенации) строк служит символ +.

Функции работы со строками

В языке Turbo Basic имеется большое число функций для работы со строками:

LEN(string)	- возвращает длину строки;
LEFT$(string,n)	- возвращает n первых символов строки;
RIGHT$(string,n)	- возвращает n последних символов строки;
MID$(string,positio,n)	- выделяет из строки n символов, начиная с указанной позиции;
MID$ в левой части выражения	- вставляет подстроку в строку;
INSTR(string,substring)	- ищет подстроку в строке;
STRING$(number,char)	- создает новую строку, сцепляя указанную подстроку n раз;
SPACE$(n)	- создает новую строку из n пробелов;
ASC(char)	- возвращает численное значение символа;
CHR$(code)	- возвращает символ, соответствующий указанному числовому значению;
STR$(n)	- преобразует число в строку символов;
VAL(string)	- преобразует строку символов в число;
HEX$(n), OCT$(N), BIN$(N)	- преобразуют число в строку шестнадцатеричных, восьмеричных и двоичных чисел соответственно;
UCASE$(string	преобразует все буквы строки в прописные;
LCASE$(string)	преобразует все буквы строки в строчные.

Данные

Данные в языке Turbo Basic (как и в Бейсике) - это тяжелое наследие того времени, когда файлов не было вообще. Сейчас это трудно представить, но файлов данных действительно не было, были только программы, внутри которых хранились данные, и чтобы модифицировать данные, нужно было модифицировать программу.

Для работы с данными применяются операторы DATA, READ и RESTORE, которые полностью эквивалентны аналогичным (и одноименным) операторам, используемым в стандартном Бейсике, и поэтому (а также в связи с тем, что они морально устарели) рассматриваться не будут.

Массивы

Описываются с помощью оператора DIM.

Формат:

DIM [STATIC | DYNAMIC] name(dim1, dim2,...)

Максимальное число размерностей массива может достигать 10; размерность может задаваться как числом, так и диапазоном чисел (например, 10:33). По умолчанию принят тип памяти STATIC. Функция FRE(-1) позволяет определить объем свободной динамической памяти. С помощью оператора ERASE можно удалить из памяти массив, объявленный как имеющий тип памяти DYNAMIC.

Формат:

ERASE name

Директивы компиляции $DYNAMIC и $STATIC указывают, какой тип памяти получают операторы DIM, следующие в программе за этими директивами.

Оператор SWAP позволяет обменять содержимое двух элементов массива.

Формат:

SWAP element1,element2

Пример:

SWAP A(1),A(15)

2.10. ФАЙЛЫ

В языке Turbo Basic работа с файлами осуществляется так же, как и в стандартном интерпретаторе GWBASIC. Файлы делятся на файлы прямого доступа, последовательные файлы и двоичные файлы.

Кроме этого, в языке Turbo Basic имеются операторы и команды, предназначенные для работы не с содержимым файлов, а с файловой средой (дисками и каталогами).

Язык Turbo Basic унаследовал эклектичность Бейсика (вызванную его многолетним развитием), которая наиболее заметна при работе с файлами. Например, операторы, предназначенные для работы с файлами разных типов, различаются синтаксически и семантически. На примере Бейсика хорошо видна необходимость жесткой стандартизации в языках программирования.

Файлы прямого доступа

В Бейсике файл прямого доступа - это файл, к N+1 компоненте которого можно обратиться, не обрабатывая перед этим N предыдущих компонент. Компоненты такого файла называются "записями". Все записи файла имеют одинаковую (фиксированную) длину. Запись можно логически разделить на более мелкие единицы - поля записи. В программе можно обращаться к отдельным полям записи, но в операции ввода-вывода всегда участвует целая запись. По-видимому, файлы прямого доступа появились в Бейсике в то время, когда на внешний носитель можно было записывать не двоичную, а только текстовую (литерную) информацию, поэтому вся информация в файлы прямого доступа записывается в виде цепочек литер (строк). В случае необходимости вывода двоичной информации программист сам должен преобразовать ее в текстовую. Поля записи представляют собой цепочки литер, расположенные в записи плотно друг за другом. При этом сумма длин полей может быть меньше длины записи.

Для работы с файлами прямого доступа используют операторы и функции OPEN, FIELD, LOF, LSET, RSET, PUT#, GET#, MKI$, MKL$, MKS$, MKD$, CLOSE, KILL, NAME. Все они тождественны аналогичным операторам и функциям интерпретатора GWBASIC.

Оператор OPEN

"Открывает" файл, т.е. делает файл на диске известным для программы, связывает имя файла на диске с номером канала ввода-вывода в программе и, при необходимости, изменяет длину записи, принятую по умолчанию.

Формат:

```
OPEN "R" #chan, filename, lrecl
OPEN filename FOR RANDOM AS #chan LEN=lrecl
```

Здесь (и далее в описании операторов):

chan - канал ввода-вывода, ассоциированный с файлом;
filename - имя файла;
lrecl - длина записи файла;
field - имя поля записи;
lenN - длина поля записи;
recnum - номер записи.

Замечание. Оба варианта формата оператора идентичны, но второй понятнее при чтении.

Пример:

```
OPEN "R", #1, "PERSON", 128
```

Оператор FIELD

Указывает структуру записи (число байт в записи, которое занимает каждое поле записи).

Формат:

```
FIELD #chan, len1 AS field1, len2 AS field2...
```

Пример:

```
FIELD #1, 12 AS NAME$, 20 AS STREET$, 16 AS CITY$
```

Теперь в записи определены три поля: NAME - длиной 12 байт, STREET - длиной 20 байт и CITY - длиной 16 байт.

Оператор LSET

Помещает значение текстовой переменной или литерала в поле записи для последующего вывода в файл. Если длина поля больше длины помещаемого в него значения, то поле дополняется пробелами справа.

Формат:

```
LSET field$=variable
```

Пример:

```
INPUT "Имя "; X$
LSET NAME$=X$
```

Замечание. Если для помещения текстовых переменных и литералов в поля записей используется простой оператор присваивания, то для помещения (и извлечения) числовой информации служит большой набор функций:

Объект	Помещение (запись)	Извлечение (считывание)
Целое число(<256)	CHR$(X)	ASC(X$)
Целое число	MKI$(X)	CVI(X$)
Длинное целое	MKL$(X)	CVL(x$)
Вещественное число	MKS$(X)	CVS(X$)
Вещественное число двойной точности	MKD$(X)	CVD(X$)

Оператор RSET

Полностью эквивалентен оператору LSET, но дополнение пробелами производится слева.

Формат:

```
RSET field$=variable
```

Функция LOF

Позволяет получить размер файла в байтах.

Формат:

```
LOF(chan)
```

Пример:

```
NUMBREC = LOF(1)/128
```

Оператор PUT

Выводит запись в файл на указанное место (в соответствии с указанным номером записи).

Формат:

```
PUT #chan, recnum
```

Пример:

```
PUT #1, RANG
```

Оператор GET

Считывает из файла запись, номер которой указан параметром recnum.

Формат:

```
GET #chan, recnum
```

Замечание. Конечно, запись с таким номером должна существовать в файле.

Пример:

```
GET #1, NNN
```

Оператор CLOSE

"Закрывает" файл, т.е. отсоединяет его от программы и освобождает в ней канал ввода-вывода. Если прекратить работу программы, не "закрыв" файл, то можно потерять информацию на диске.

Формат:

CLOSE #chan, chan...

Пример:

CLOSE #1, #2

Сервисные операции с файлами

В языке Turbo Basic имеются предназначенные для работы с файлами операторы, не изменяющие их содержимого. К ним относятся операторы NAME, KILL, FILES, MKDIR, CHDIR, RMDIR.

Оператор NAME

Предназначен для переименования файлов на диске.

Формат:

NAME filename1 AS filename2

Пример:

NAME "PERSON" AS "PERSON2"

Оператор KILL

Удаляет файл с диска.

Формат:

KILL filename

Пример:

KILL "B: SAMPLE. BAS"

Оператор FILES

Позволяет получить на экране список файлов, находящихся на диске.

Формат:

FILES wildcard

Пример:

FILES "*. EXE"

Замечание. Очевидно, что этот оператор следует применять в режиме непосредственного исполнения, а не в программе.

Оператор MKDIR

Позволяет создать на диске каталог.

Формат:

MKDIR path

Пример:

MKDIR "\CAT1"

Оператор CHDIR

Позволяет сменить текущий каталог.

Формат:

CHDIR path

Пример:

CHDIR "\CAT1"

Оператор RMDIR

Удаляет указанный каталог с диска.

Формат:

RMDIR path

Пример:

RMDIR "\CAT1"

Замечание. Конечно, удаляемый каталог должен быть пустым.

Совместимость с GSBASIC

Формат чисел в файлах, созданных системой Turbo Basic, не совпадает с форматом чисел, созданных интерпретатором GWBASIC. Для обеспечения совместимости следует вместо функций CVI, CVS, CVD, MKI$, MKS$, MKD$ пользоваться функциями CVMI, CVMS, CVMD, MKMI$, MKMS$, MKMD$ соответственно.

Последовательные файлы

Последовательные файлы следует рассматривать как замену экрана и клавиатуры. Все, что выводилось на экран, в таком же виде и по таким же правилам выводится в последовательный файл, а все, что вводилось с клавиатуры, вводится из последовательного файла.

Для работы с последовательными файлами используют операторы и функции OPEN, PRINT#, INPUT#, LINE INPUT #, INPUT$, CLOSE, EOF(), WRITE, тождественные аналогичным операторам и функциям интерпретатора GWBASIC.

Оператор OPEN

"Открывает" файл, т.е. делает файл на диске известным для программы, связывает имя файла на диске с номером канала ввода-вывода в программе и указывает режим работы с файлом: O (OUTPUT) - вывод, I (INPUT) - ввод, A (APPEND) - добавление в конец файла.

Формат:

OPEN "I", #chan, filename
OPEN "O", #chan, filename
OPEN "A", #chan, filename
OPEN filename FOR INPUT AS #chan
OPEN filename FOR OUTPUT AS #chan
OPEN filename FOR APPEND AS #chan

Здесь (и далее в описании редакторов):

chan - канал ввода-вывода, ассоциированный с файлом;
filename - имя файла;
variable - переменная;
string - строковая переменная.

Замечание. Как и в случае файлов прямого доступа, оба варианта формата оператора идентичны.

Пример:

OPEN "O", #1, "SALARIES"

Оператор PRINT#

Предназначен для вывода литерного представления значения указанной переменной в файл. Образно говоря, этот оператор "печатает в файл". После значения переменной выводятся символы "Возврат каретки" и "Перевод строки".

Формат:

PRINT #chan, variable

Пример:

PRINT #1, NAME$
PRINT #1, SALARY

Оператор INPUT#

Предназначен для ввода значений переменных из файла (так же, как это делает оператор INPUT при их вводе с клавиатуры).

Формат:

INPUT #chan, variable1, variable2,...

Пример:

INPUT #1, NAME$

Замечание. Чтобы с помощью оператора INPUT# можно было ввести несколько переменных, их следует разделить запятыми. Ввод прекращается при появлении символа "Возврат каретки".

Функция EOF

Позволяет определить конец файла.

Формат:

EOF(chan)

Оператор CLOSE#

"Закрывает" файл, т.е. отсоединяет его от программы и освобождает в ней канал ввода-вывода.

Формат:

CLOSE #chan, chan...

Пример:

CLOSE #1, #2

Оператор LINE INPUT#

Как и оператор INPUT#, аналогичен оператору LINE INPUT, предназначенному для ввода данных с клавиатуры; позволяет вводить данные, содержащие запятые.

Формат:

LINE INPUT #chan, string

Пример:

LINE INPUT #1, RECORD$

Функция INPUT$

Служит для ввода из файла указанного числа байт.

Формат:

INPUT$(number of chars, #chan)

Пример:

C$=INPUT$(2, #1)

Оператор WRITE#

Аналогичен оператору PRINT#; предназначен для вывода переменных в файл, но строковые переменные должны быть заключены в кавычки, а все переменные следует разделять запятыми.

Формат:

WRITE #chan, variable1, variable2,...

Пример:

WRITE #1, NAME$, STREET$

Замечание. Формат вывода оператора WRITE# - это именно тот формат, который требуется для ввода данных в таких пакетах, как dBase III и Lotus 1-2-3.

Двоичные файлы

Определение файла как двоичного позволяет работать с ним, игнорируя его структуру, и допускает обращение к любому байту файла независимо от того, является ли содержимое этого байта запятой, символом "Возврат каретки" или еще каким-либо специальным символом.

При работе с двоичным файлом в системе Turbo Basic постоянно ведется указатель местонахождения в файле - попросту отслеживается номер текущего байта.

Для работы с двоичными файлами используют операторы OPEN, PUT$, GET$, SEEK, CLOSE и функции LOC и LOF.

Оператор OPEN

Наверно, читатель уже запомнил, что этот оператор "открывает" файл.

Формат:

OPEN "B", #chan, filename
OPEN filename FOR BINARY AS #chan

Замечание. Оба варианта формата оператора идентичны.

Пример:

OPEN "B", #1, "B: FBIN"

Оператор PUT$

Предназначен для вывода строки символов в двоичный файл.

Формат:

PUT$ #chan, string

Пример:

PUT$ #1, "ABCDE"

Оператор GET$

Предназначен для ввода некоторого числа байт из двоичного файла.

Формат:

GET$ #chan, number of bytes, string

Пример:

GET$ #1, 5, X$

Оператор SEEK

Используется для установки указателя на нужный байт в файле.

Формат:

SEEK #chan, byte address

Пример:

SEEK #1, 3

Оператор CLOSE

"Закрывает" файл.

Формат:

CLOSE chan

Пример:

CLOSE #1

Функция LOC

Указывает текущую позицию в файле.

Формат:

LOC(chan)

Функция LOF

Позволяет определить конец файла.

Формат:

LOF(chan)

2. 11. ЦВЕТ И МУЗЫКА

Turbo Basic обладает богатыми графическими и музыкальными возможностями, во многом совпадающими с возможностями интерпретатора GWBASIC. Это позволяет сократить их рассмотрение.

Оператор SCREEN

Определяет графический режим и использование палитры.

Формат:

SCREEN mode[,color]

Режим работы адаптера связан с числовым значением параметра mode следующим образом:

1 - CGA (320 x 200);
2 - CGA (640 x 200) и Hercules (720 x 348);
7 - EGA (320 x 200), 16 цветов;
8 - EGA (640 x 200), 4 цвета или 16 цветов из 16;
9 - EGA (640 x 350), 4 цвета или 16 цветов из 64;
10 - EGA (640 x 350), режим демонстрации монохромного изображения.

Параметр color имеет значение 0 (монохромное изображение) или 1 (цветное изображение).

Оператор COLOR

Определяет используемые фон и палитру для графического режима.

Формат:

COLOR background, palette

Графические операторы

Делятся на три группы: PSET, LINE, CIRCLE (рисование с помощью точек, прямых и окружностей); DRAW (рисование "пером"); GET и PUT (анимация).

Рисование точек, прямых и окружностей

Форматы:

PSET (x, y), color	- установка цветной точки по абсолютному адресу;
PSET STEP (dx, dy)	- установка цветной точки по адресу, вычисленному относительно адреса предыдущей точки;
PRESET (x, y)	- гашение точки, имеющей абсолютный адрес;
PRESET STEP (dx, dy)	- гашение точки, имеющей относительный адрес;
LINE (x1, y1)-(x2, y2), color	- рисование линии, начало и конец которой заданы абсолютными адресами;
LINE STEP (dx1, dy1)-(dx2, dy2), color	- рисование линии, начало и конец которой заданы относительными адресами;
CIRCLE (xc,-yc), radius, color	- рисование окружности;
POINT(x, y)	- получение цвета точки с указанными адресами.

Рисование "пером"

Формат:

DRAW string - описание движения пера.

Строка string содержит "команды для пера". Команды, разделенные пробелами, идут одна за другой. В команде должны быть указаны первая буква слова, обозначающего направление, и смещение, измеряемое в точках (пикселах). Буквы имеют следующие значения:

U (up) - вверх;	H - вверх и влево;
D (down) - вниз;	G - вниз и влево;
L (left) - влево;	E - вверх и вправо;
R (right) - вправо;	F - вниз и вправо.

Кроме описанных выше относительных координат могут быть заданы абсолютные координаты (в виде M130) и цвета (в виде C3, где 3 - номер цвета).

Анимация

Сначала графические изображения чертят обычными способами, затем они запоминаются в таблицах, быстрая

смена которых на экране создает эффект мультипликации.

Форматы:

```
GET(x1, y1)-(x2, y2), table
PUT(x, y), table, type
```

Здесь:
xN и yN - координаты;
type - тип операции наложения нового изображения на имеющееся на экране:

```
XOR
PSET
PRESET
OR
AND
```

Пример:

```
SCREEN 1
CLS
DIM T(4)
LINE (0, 0)-(7, 7)
GET (0, 0)-(7, 7), T
PUT (100, 100), T
```

Музыка

Музыкальные возможности языка Turbo Basic реализуются при использовании двух простых операторов BEEP и SOUND и мощного оператора PLAY.

Форматы:

```
BEEP number
SOUND freq, ticks
PLAY string
```

При выполнении первого оператора динамик ПК издает звук частотой 800 Гц в течение указанного интервала времени, измеряемого в 1/4 с. При выполнении второго оператора динамик ПК издает звук заданной частоты (от 37 до 32767 Гц) в течение интервала времени, измеряемого в 1/18 с. В третьем операторе (в параметре string) можно записать практически любое музыкальное произведение, пользуясь обычной нотной записью. Размер строки ограничен буфером музыкальной подсистемы.

Применяемая система обозначений (приведена не полностью, поскольку многократно описана, в том числе в рекомендованной литературе):

- ноты DO, RE, MI, FA, SOL, LA, SI кодируются буквами C, D, E, F, G, A, B;
- октава кодируется буквой O, за которой следует номер октавы (например, O2);
- длительность кодируется как Lnn, где nn изменяется от 1 до 64 (1 означает нормальную длительность, 64 - длительность 1/64);
- темп кодируется как Tnn, где nn изменяется от 32 до 255 (по умолчанию принято значение 120);
- пауза кодируется как Pn, где n изменяется от 1 до 64;
- абсолютное значение ноты Nn обозначает одну из 84 нот.

2. 12. ДАТА И ВРЕМЯ

Функция DATE$

Позволяет узнать дату, хранящуюся в операционной системе, а при употреблении функции в левой части выражения эту дату можно переопределить.

Пример:

```
PRINT DATE$
D$="05-03-90"
PRINT D$
```

Результат (в предположении, что было 5 февраля):

```
05-02-90
05-03-90
```

Функция TIME$

Полностью аналогична функции DATE$. Позволяет узнать и переопределить хранящееся в операционной системе время.

Формат:

```
05: 07: 00
```

Функция TIMER

Позволяет узнать время в секундах.

Формат:

```
N=TIMER()
```

Оператор DELAY

Приостанавливает выполнение программы на заданное число секунд.

Формат:

```
DELAY secs
```

2.13. СПЕЦИАЛЬНО ДЛЯ ХЭКЕРОВ

В этом (тринадцатом!) разделе описаны возможности языка Turbo Basic, которые не нужны простому прикладному программисту (я хотел бы увидеть системного программиста или хэкера, который пишет программы на Бейсике), но тем не менее эти возможности в языке имеются.

Прямой доступ к памяти ПК

Прямой доступ позволяет прочесть или изменить содержимое любой ячейки памяти ПК из программы на Бейсике (зачем только это нужно делать?).

Для этого сначала следует с помощью оператора DEF SEGMENT определить сегмент (т.е. группу из 16 байт).

Формат:

```
DEF SEGMENT=segment
```

Здесь segment задается в виде числа, например 64.

Затем с помощью функции PEEK можно прочесть содержимое ячейки памяти, а с помощью оператора POKE изменить его.

Форматы:

```
N=PEEK(addr)
POKEaddr, value
```

Здесь:

addr - адрес внутри сегмента;
value - новое значение.

Запись содержимого ячейки памяти на диск и чтение его с диска

Можно записать на диск в специальном формате содержимое участка памяти ПК и потом ввести его с диска в тот же или в другой участок памяти. Это делается с помощью операторов BSAVE и BLOAD.

Форматы:

```
BSAVE filename, addr, length
BLOAD filename, addr
```

Здесь:

filename - имя файла;
fddr - адрес начала участка памяти;
length - длина участка памяти.

Следующая программа предназначена для записи на диск содержимого экрана и последующего его считывания (вот как хранить "красивые экраны"):

```
DEF SEG=&HB800
BSAVE "screen", 0, &H4000
...
DEF SEG=&HB800
BLOAD "screen", 0
```

Получение адреса переменной

Иногда необходимо узнать, где в памяти ПК расположена та или другая переменная. Это можно сделать с помощью функций VARSEG и VARPTR.

Форматы:

```
Variable=VARSEG(X)
Variable=VARPTR(x)
```

Первая функция возвращает значение сегмента, а вторая - значение смещения в сегменте.

Работа с портами

Работа с портами осуществляется с помощью функции INP, предназначенной для ввода из порта, и оператора OUT, предназначенного для вывода в порт.

Форматы:

```
Variable=INP(port)
OUT port, byte
```

Параметр port задается как целое.

Вызов прерывания DOS

Сначала с помощью оператора REG надо установить значения регистров, а затем с помощью оператора CALL INTERRUPT вызвать нужное прерывание.

Форматы:

```
REG register, value
CALL INTERRUPT Intnumber
```

Здесь register - номер регистра:

0 - флажки	5 - SI
1 - AX	6 - DI
2 - BX	7 - BP
3 - CX	8 - DS
4 - DX	9 - ES

ЗАКЛЮЧЕНИЕ

Конечно, в рамках журнальной публикации трудно подробно рассказать обо всех возможностях такого мощного языка, как Turbo Basic, однако автор надеется, что даже столь краткое описание поможет программистам.



ПРЕДЛАГАЕТ ДЛЯ ПЕРСОНАЛЬНЫХ КОМПЬЮТЕРОВ, СОВМЕСТИМЫХ С IBM PC:

МУЛЬТИИНТЕРФЕЙСНЫЕ КАРТЫ П1, П1Т, П2,

предназначенные для организации ввода-вывода цифровой информации:

П1

24 двунаправленные линии ввода-вывода, четыре управляющих сигнала;

П1Т

24 двунаправленные линии, три 16-разрядных таймера, кварцевый генератор, четыре управляющих сигнала;

П2

48 двунаправленных линий ввода-вывода, четыре управляющих сигнала.

ПРИЛАГАЕМЫЕ ПРОГРАММНЫЕ СРЕДСТВА позволяют:

организовать любой протокол обмена между ЭВМ и периферийным оборудованием;

оперативно проводить тестирование и управление аппаратными средствами

(датчиками, измерительными приборами, ЦАП, АЦП, мультиплексорами и т.д.).



КОЛЬЦЕВУЮ ЛОКАЛЬНУЮ ВЫЧИСЛИТЕЛЬНУЮ СЕТЬ С-1

- Физическая скорость обмена информацией между абонентами 500 Кбит/с.
- Интегрированная скорость обмена информацией 6 Кбит/с.
- До 100 подключаемых абонентов.
- 64 одновременно устанавливаемых и поддерживаемых виртуальных канала (для каждого абонента).



Приобрести изделия можно в магазинах-салонах "Приборы и вычислительная техника" в городах Москва (129-09-36), Минск (34-62-56), Киев (261-31-86).
Наш московский телефон:
284-48-51, 124-62-60 (автоответчик).

АНТИВИРУСЫ



В.Б. КОМЯГИН

Антивирусная программа VP-Virus Protector

Рассматривается антивирусная программа-монитор, предназначенная для защиты компьютеров от заражения вирусами.

Антивирусная программа VP (Virus Protector), резидентно размещаемая в оперативной памяти компьютера, позволяет информировать пользователя о подозрительных действиях других программ, предлагая подтвердить их необходимость. Эта программа помогает избежать распространения "инфекции" по всему жесткому диску зараженного компьютера, но она не обеспечивает 100%-ной защиты от вирусов и не способна помочь в случае поражения загрузчика операционной системы. Однако идеи, использованные при написании модели этой программы, могут послужить основой для создания ее более мощной версии.

Антивирусная программа VP выполняет следующие функции:

- ограничивает доступ к файлам, указанным поименно, или группам файлов по расширениям;
- сообщает пользователю о попытках программ закрепиться в памяти;
- проверяет правильность назначения некоторых векторов прерываний;
- защищает от записи нулевые секторы всех дисков.

Следует подчеркнуть, что программа VP реализует режим разделения времени и ее работа практически не сказывается на времени выполнения основной программы.

При каждой попытке изменить защищаемые объекты (файлы, векторы прерываний, секторы, память) программа VP предлагает пользователю подтвердить необходимость этих действий, т.е. позволяет работать в режиме "авторизованного доступа". В настоящее время разработано много подобных программ для реализации этого режима.

Отличительными особенностями предлагаемой программы являются:

1. *Ненавязчивость*. Большинство антивирусных программ-мониторов ограничивают доступ ко всем выполняемым файлам типа EXE или COM. Если в процессе работы создается много подобных файлов, то необходимость в подтверждении доступа возникает очень часто, что мешает оператору. Программа VP ведет себя в этом случае "ненавязчиво" благодаря тому, что защищает не все выполняемые файлы, а лишь некоторые "жизненно важные" файлы или файлы-ловушки, к которым заведомо не должно быть обращений.

2. *Организация режима разделения времени*. Для проверки назначения векторов прерываний программа VP организует режим разделения времени. При переназначении вектора прерывания имеется возможность восстановить его предыдущее значение.

Рассмотрим исходный текст программы VP, приведенный в этой статье. Стартовый адрес установлен на 100h (строка 3), что позволяет создать выполняемый файл типа COM. Далее следуют переход на инициализацию (строка 5) и исходные данные, необходимые для работы программы.

Таблица имен файлов, которые программа VP защищает от несанкционированного доступа, состоит из двух частей: первая часть (строки 89 - 96), содержащая шаблоны имен файлов, позволяет в процессе трансляции основной программы формировать указатели на концы этих шаблонов; вторая часть таблицы (строки 96 - 103) содержит длины шаблонов имен файлов и указатели на концы этих шаблонов, приведенных в первой части. Признак конца таблицы - нулевое слово (строка 103).

Ввиду ограниченного объема статьи ниже приведена версия программы VP, которая контролирует доступ к файлам, осуществляемый только с помощью дескриптора File Handle. Операции, использующие для этой цели блок FCB (File Control Block), игнорируются. Однако, разобравшись в исходном тексте предлагаемой программы, Вы сможете легко усовершенствовать ее, добавив контроль доступа к файлам, осуществляемого с помощью этих операций.

В случае использования дескриптора File Handle запрос на открытие файла содержит ASCIZ-строку имени файла (строку ASCII-символов,

заканчивающуюся нулевым байтом). Программа VP перехватывает этот запрос по вектору прерывания 21h и ведет поиск подходящего шаблона в таблице TableName (строки 287 - 301). Сравнение строк шаблона и имени файла организовано с конца, поэтому контролировать можно или только расширения файлов, или их полные имена, включающие весь путь доступа (строки 198 - 218).

При совпадении расширения файла или его имени с одним из шаблонов, например с COMMAND.COM, выполнение основной программы прерывается и на экране появляется окно с вопросом: “Вы хотите открыть файл COMMAND.COM?” и меню с двумя альтернативами: “ Да” или “Нет”. Если Вы ответите утвердительно, файл будет открыт; в противном случае в регистре ah появится код ошибки “Доступ запрещен”. Аналогично осуществляется контроль переназначения векторов:

- 8h - прерывания от таймера;
- 13h, 26h и 25h - чтение/запись с диска;
- 21h - функции DOS.

По существу программа VP производит мониторинг использования прерывания 27h и функции 31h перывания 21h, а также контроль попыток перезаписать загрузочный сектор диска с помощью прерываний 13h и 26h.

В тексте программы VP широко используются макросы. Следует обратить внимание на макрос ChangeHandler, так как он позволяет генерировать имена меток по номеру Vector, например из строки mov Old&Vector&o, bx при подстановке вместо Vector номера 21 генерируется строка mov Old21o, bx. Подход, применяемый в макросе ChangeHandler для изменения адреса обработчика прерываний, основан на использовании самомодифицирующегося кода.

Каждая процедура обработки прерывания заканчивается переходом на первоначальный обработчик прерываний - инструкцией jmp far 0000:0000, закодированной с помощью директив db и dw. Местонахождение сегмента перехода и его смещение определяются в момент переназначения вектора прерывания (например, строки 697 - 698).

Рассмотрим самую интересную часть программы VP - организацию работы в режиме разделения процессорного времени с помощью аппаратного прерывания от таймера по вектору 8h. Выполнение основной программы прерывается приблизительно каждые 165 мс, и управление получает процедура Protector, которая сравнивает текущие адреса обработчиков прерываний с их первоначальными значениями в момент загрузки антивирусной программы VP или последнего санкционированного изменения выполняемой программы.

Передача управления процедуре Protector осуществляется таким образом. При возникновении прерывания по вектору 8h процессор заносит в стек содержимое следующих регистров прерванной программы: регистра флагов, регистров CS и IP. Далее управление передается программе VP на метку New8 (строка 316). Если время работы основной программы, определяемое переменной Quant, исчерпано, для приостановки дальнейшего отсчета времени переключается семафор Switch (строка 320), сохраняется содержимое всех регистров основной программы в стеке и далее в стек помещается содержимое регистра флагов, содержимое сегментного регистра программы VP и адрес процедуры Protector. Затем, при возврате из прерывания по инструкции iret, управление передается процедуре Protector. После того как процедура Protector выполнила свои действия, восстанавливается содержимое регистров прерванной программы (строка 344), освобождается семафор Switch и управление передается прерванной программе с помощью инструкции iret, так как в стеке хранится содержимое регистров IP, CS и регистра флагов прерванной программы.

Работа программы VP начинается с проверки, не допускающей ее повторной загрузки (строки 687-691), затем переназначаются векторы прерываний, необходимые для работы программы VP, и после этого адреса обработчиков прерываний запоминаются в таблице имен файлов. Следует обратить внимание на то, что переназначение вектора прерывания 8h происходит в последнюю очередь. Выход из программы VP и размещение ее резидентно в памяти осуществляется по прерыванию int 21h (функция 31h), управление на обработчик прерывания передается с помощью инструкции jmp, что позволяет предотвратить перехват его самой программой VP.

Несколько слов о возможностях совершенствования описываемой модели антивирусной программы. В режиме разделения времени можно контролировать попытки остаться резидентными в памяти тех выполняемых программ, которые не используют для этого запросы BIOS или DOS. Интересной представляется возможность ведения на диске (без буферизации в памяти) протокола изменений файловой системы, содержащего сведения о том, какая программа какие файлы модифицировала. Этот протокол позволит установить источник “заражения” компьютера и предотвратить распространение “инфекции” таким образом, чтобы “зависание” компьютера из-за деятельности вирусов не приводило к потере ценной для анализа информации.

Текст программы

```
1    Code    Segment
2    Assume  cs:Code, ds:Code, es:Code
3    org     100h
4    Start:
5            jmp     init
6    ;
7    ShadeAtr        =       04h     ;Атрибут тени окна...
8    CurrentAtr      =       4fh     ;Цвет символов + цвет фона окна...
9    ;флаги процедуры "AskUser"...
10   AskAboutFile    =       1  ;Задать вопрос о файле...
11   AskAboutVector  =       2  ;Задать вопрос о векторе...
12   AskAboutMemory  =       3  ;Задать вопрос о памяти...
13   AskAboutSector  =       4  ;Задать вопрос о секторе...
14   ;флаги процедуры "BoxHandler"...
15   InversFunction  =       1  ;Поменять местами цвета фона и символов...
16   ShadeFunction   =       2  ;Нарисовать тень окна...
17   PutFunction     =       3  ;Вывести окно или строку на экран...
18   RestoreFunction =       4  ;Восстановить участок экрана...
19   SaveFunction    =       5  ;Сохранить участок экрана...
20   ;
21   EnterKey        =       13 ;Код клавиши возврата каретки...
22   ;
23   Window          db  "                                                     "
24                   db  " ****************** Virus Protector ****************** "
25                   db  " *                                                 * "
26                   db  " *                                                 * "
27                   db  " *                                                 * "
28                   db  " *    Вы уверены ?       Да      Нет              * "
29                   db  " *************************************************** "
30                   db  "                                                     "
31   ;Слово - координаты верхнего левого угла окна на экране...
32   WindowPos       =       12 + 7 * 256 ;12-я колонка, 7-я строка...
33   ;Слово - размер окна: 55 колонок, 8 строк...
34   WindowSize      =       55 + 8 * 256
35   ;Координаты и размеры меню "Да-Нет"...
36   YesPos          =       34 + 12 * 256
37   YesLen          =       4
38   NoPos           =       40 + 12 * 256
39   NoLen           =       5
40   ;
41   VecMsg          db      "Вы хотите переназначить вектор ?"
42   VecMsgLen       =       32
43   VecMsgPos       =       20 + 10 * 256
44   Vec8Msg         db      "08h" ?".
45   Vec13Msg        db      "13h" ?"
46   Vec25Msg        db      "25h" ?"
47   Vec26Msg        db      "26h" ?"
48   Vec21Msg        db      "21h" ?"
49   VecXMsgPos      =       52 + 10 * 256
50   VecXMsgLen      =       6
51   ;
52   FileMsg         db      "Вы хотите открыть файл ?"
53   FileMsgPos      =       28 + 9 * 256
54   FileMsgLen      =       22
55   ;
56   SecMsg          db      "Вы хотите переписать загрузочный сектор ?"
57   SecMsgLen       =       41
58   SecMsgPos       =       20 + 10 * 256
59   ;
60   MemMsg          db      "Вы хотите оставить резидентной программу ?"
61   MemMsgLen       =       40
62   MemMsgPos       =       21 + 9 * 256
63   ;
64   ErrMsg          db       с неизвестным именем
65   ErrMsgLen       =       20
66   ;Буфер для сохранения участка экрана: <+2> и <+1> для тени...
67   SaveBuffer      db      (( 55 + 2 ) * ( 8 + 1 ) * 2) dup(?)
68   ;Буфер для сохранения позиции курсора и его размера...
69   OldCursorPos    dw      ?
70   OldCursorShape  dw      ?
71   ;Таблица первоначальных адресов обработчиков прерываний...
72   OldVectorTable  label   word
73   Vector8         dd      ?
74   Vector13        dd      ?
```

```
Vector25            dd   ?
Vector26            dd   ?
Vector21            dd   ?
;Таблица текущих адресов обработчиков прерываний...
NewVectorTable      label      word
Vector_8            dd   ?
Vector_13           dd   ?
Vector_25           dd   ?
Vector_26           dd   ?
Vector_21           dd   ?
;Число квантов времени, в течение которых работает фоновая программа...
Quant               db   3
;Семафор: Switch=0 - работает VP, Switch=1 - работает фоновая программа...
Switch              db   1
;Таблица шаблонов имен файлов, которые защищает VP...
                    db   'COMMAND.COM'
FileName1           db   0
                    db   '.EXE'
FileName2           db   0
                    db   'V-P.ASM'
FileName3           db   0
;
TableName           dw   11  ;длина строки имени файла "COMMAND.COM"...
                    dw   FileName1 ;указатель на конец имени файла...
                    dw   4 ;длина строки ".EXE"...
                    dw   FileName2 ;указатель на конец шаблона имени файла...
                    dw   5
                    dw   FileName3
                    dw   0 ;признак конца таблицы...
;Сохранить содержимое регистров в стеке...
PushReg             macro    reg
                    irp  r,<reg>
                    push    r
                    endm
endm
;Восстановить содержимое регистров из стека...
PopReg              macro    reg
                    irp  r,<reg>
                    pop  r
                    endm
endm
;Выполнить видеофункцию с 16-ричным номером "fun"...
Video               macro    fun
                    mov  ah,fun&h
                    int  10h
                    endm
;Выполнить операцию с участком экрана...
Box                 macro    Msg,MsgPos,MsgSize,Function
                    ifidn <si>,<Msg>
                              ;Ничего не делать...
                    else
                          lea          si,Msg
                    endif
                    ifidn <dx>,<MsgPos>
                              ;Ничего не делать...
                    else
                          mov          dx,MsgPos
                    endif
                    ifidn <cx>,<MsgSize>
                              ;Ничего не делать...
                    else
                          mov          cx,MsgSize
                    endif
                    mov  di,Function
                    call BoxHandler
                    endm
;Инструкция MOV для сегментных регистров...
MoveSeg             macro    Seg1,Seg2
                    push     Seg2
                    pop  Seg1
                    endm
;Поместить адрес обработчика вектора прерывания с 16-ричным
;номером "Vector" в позицию таблицы TablePos...
PutIntoTable        macro    Vector,TablePos
                    mov  ax,ds:[ Vector * 4 ]
                    mov  es:[ di + TablePos ],ax
                    mov  ax,ds:[ Vector * 4 + 2 ]
                    mov  es:[ di + TablePos + 2 ],ax
                    endm
;Восстановить первоначальный адрес обработчика вектора прерывания
;с 16-ричным номером "Vector" из позиции таблицы AbsPos...
TakeOutTable        macro    Vector,AbsPos
                    mov  ax,es:[ di + AbsPos ]
                    mov  ds:[ Vector * 4 ],ax
                    mov  ax,es:[ di + AbsPos + 2 ]
                    mov  ds:[ Vector * 4 + 2 ],ax
                    endm
;Установить новый обработчик вектора прерывания
;с 16-ричным номером "Vector", сохранив старый адрес...
ChangeHandler       macro    Vector
                    mov  ax,35&Vector&h
                    int  21h
                    mov  old&Vector&o,bx
                    mov  old&Vector&s,es
                    lea  dx,New&Vector
                    mov  ax,25&Vector&h
                    int  21h
                    endm
;**************************************************************
;  Вычислить длину строки и сделать все буквы заглавными.
;  вход:  ds:si - указатель на ASCIZ строку.
;  выход:  ax    - длина строки в байтах.
;**************************************************************
StrLen              proc       near
                    xor  cx,cx
next_char:
                    mov  al,ds:[si ]  ;B al текущий символ...
                    cmp  al,0 ;Это конец строки ?
                    jz   end_string ;Да...
                    inc  cx ;Увеличить содержимое счетчика символов...
                    cmp  al,"a" ;Проверить: ("a" <= al <= "z") ?
                    jl   no_lower
                    cmp  al,"z"
                    jg   no_lower
                    sub  al,' ' ;Сделать все буквы заглавными...
                    mov  ds:[si],al
no_lower:
                    inc  si ;Указатель на следующий символ...
                    jmp  next_char
end_string:
                    mov  ax,cx ;B регистре ax длина строки...
                    ret
StrLen              endp
;**************************************************************
; Сравнить две строки начиная с конца.
; вход: es:di - указатель на конец первой строки.
;          ds:si - указатель на конец второй строки.
;               ax - длина первой строки.
;               cx - длина второй строки.
; выход: флаг CARRY установлен, если строки одинаковы.
;**************************************************************
Compare             proc       near
                    cmp  ax,cx ;Сравнить длины строк...
                    jl   not_equ ;Если 1-я строка короче 2-й, то они не равны...
                    inc  cx
                    std
                    repe       cmpsb ;Сравнить строки длиной cx-1 символов...
                    jnz  not_equ
                    stc
                    ret
not_equ:
                    clc
                    ret
Compare             endp
;**************************************************************
;  Вывести вопрос: "Вы хотите оставить программу резидентной ?"
;**************************************************************
MemoryControl       proc       near
                    PushReg  <ax,dx>
                    call FindName ;Найти имя программы, выдавшей запрос TSR...
                    mov  di,AskAboutMemory
                    call AskUser ;Принять решение: "Оставить или нет ?"...
                    PopReg   <dx,ax>
                    ret
MemoryControl       endp
;**************************************************************
; Вывести вопрос: "Вы хотите переписать загрузочный сектор?"
;**************************************************************
SectorControl       proc       near
                    PushReg  <ax,bx,cx,dx,ds,es,di,si>
                    mov  di,AskAboutSector
                    call AskUser
                    PopReg   <si,di,es,ds,dx,cx,bx,ax>
                    ret
SectorControl       endp
;*****************************************
; Контроль записи в загрузочный сектор.
;*****************************************
New13:              cmp  ah,3 ;Это функция записи ?
                    jnz  old13 ;Нет...
                    cmp  cx,0 ;Запись в 0-й сектор 0-й дорожки ?
                    jnz  old13 ;Нет...
                    call SectorControl ;Принять решение: "Записывать или нет ?"
                    jnc  Old13 ;Решение = "записывать"...
                    mov  ah,3 ;Иначе: код ошибки в ah "защита записи"...
                    iret  ;Вернуться из обработчика прерывания...
old13:              db   0eah ;Первый байт инструкции JMP FAR old13s:old13s,
old13o              dw   0 ;передать управление старому обработчику...
old13s              dw   0
;**************************************************************
; Контроль записи в загрузочный сектор, (аналогично New13)."
;**************************************************************
New26:              cmp  dx,0
                    jnz  old26
                    call SectorControl
                    jnc  old26
                    mov  ah,3
                    iret
old26:              db   0eah ;Первый байт инструкции JMP FAR old26s:old26s,
old26o              dw   0 ;передать управление старому обработчику...
old26s              dw   0
;*********************************************
; Контроль попыток "остаться резидентной".
;*********************************************
New27:              call MemoryControl ;Спросить: "Оставлять или нет ?"
                    jnc  old27 ;Ответ = "оставлять"...
                    mov  ax,4c00h ;Иначе выдать DOS запрос
                    jmp  old21 ;закончить работу программы...
old27:              db   0eah ;Первый байт инструкции JMP FAR old27s:old27s,
old27o              dw   0 ;передать управление старому обработчику...
old27s              dw   0
;*************************
; Контроль DOS-запросов.
;*************************
New21:              cmp  ah,31h ;Это запрос "остаться резидентной" ?
                    jnz  next_function ;Нет...
                    call MemoryControl ;Спросить: "Оставлять или нет ?"
                    jnc  old21 ;Ответ = "оставлять"...
                    mov  ax,4c00h       ;Иначе завершить работу программы...
next_function:      cmp  ah,03dh ;Попытка открыть файл ?
                    jnz  old21 ;Нет...
                    PushReg   <ax,bx,cx,dx,di,si,es,ds>
                    mov  si,dx
                    call StrLen ;Получить в регистре ax длину имени файла...
                    lea  bx,cs:TableName ;Искать имя в таблице шаблонов...
                    MoveSeg  es,cs
next_name:          mov  si,dx ;si - указывает на начало имени файла...
```

```
                add     si,ax ;si - указывает на конец имени файла...
                mov     cx,es:[ bx ] ;cx = длина шаблона...
                mov     di,es:[ bx + 2 ] ;di = указатель на конец шаблона...
                call    Compare
                jc      no_access ;Обнаружено совпадение...
                add     bx,4 ;Иначе - следующая запись в таблице шаблонов...
                cmp     word ptr es:[ bx ],0 ;Это конец таблицы ?
                jnz     next_name ;Нет...
                clc     ;В таблице нет похожего шаблона...
                jmp     pop_all
no_access:      mov     di,AskAboutFile ;Спросить: "Открывать или нет ?"
                call    AskUser
pop_all:        PopReg  <ds,es,si,di,dx,cx,bx,ax>
                jnc     old21 ;Если имя не найдено или ответ = "открывать"...
                mov     ax,5 ;Иначе код ошибки "доступ запрещен"...
                iret
old21:          db      0eah ;Первый байт инструкции JMP FAR old21s:old21s,
old21o          dw      0 ;передать управление старому обработчику...
old21s          dw      0
;С помощью этого кода VP предотвращает повторную загрузку в память...
VP_sign         dw      5555h
;*********************************
; Обработка прерывания от таймера.
;*********************************
New8:           test    byte ptr cs:Switch,1 ;Работает фоновая программа ?
                jz      old_handler ;Нет...
                dec     cs:Quant ;Иначе уменьшаем счетчик квантов времени...
                jnz     old_handler
                mov     byte ptr cs:Switch,0 ;Переключиться на работу VP...
                PushReg <ax,bx,cx,dx,si,di,ds,es>
                pushf           ;Занесем в стек регистр флагов
                push    cs ;сегментный регистр
                lea     cx,Protector
                push    cx ;и адрес процедуры "Protector"...
;Теперь при возврате из прерывания в старом обработчике по инструкции
;IRET произойдет передача управления на процедуру "Protecror"...
old_handler:    db      0eah ;Первый байт инструкции JMP FAR old8s:old8s,
old8o           dw      0 ;передать управление старому обработчику...
old8s           dw      0
;*****************************************
; Контроль назначений векторов прерываний.
;*****************************************
Protector:      MoveSeg es,cs
                cld
                lea     di,es:NewVectorTable ;Загрузим в NewVectorTable текущие
                call    GetVectors ;адреса обработчиков прерываний...
                MoveSeg ds,cs
                lea     si,OldVectorTable
                mov     cx,10
                repe    cmpsw ;Сравним со старыми адресами...
                jnz     restore ;Обнаружены различия...
exit:           mov     byte ptr cs:Quant,3 ;Восстановить число квантов времени...
                PopReg  <es,ds,di,si,dx,cx,bx,ax>
                mov     byte ptr cs:Switch,1 ;Разрешить работу фоновой программы...
                iret    ;Вернуться в фоновую программу...
restore:        mov     di,AskAboutVector ;Спросить:
                call    AskUser ;"Переназначать вектор или нет ?"
                jnc     one_can_chang ;Ответ = "переназначать"...
                call    RestoreVectors ;Иначе восстановить старое значение...
                jmp     exit
one_can_chang:  lea     di,es:OldVectorTable
                call    GetVectors ;Загрузим новую эталонную таблицу,
                jmp     exit ;т.е. сохраним разрешенные изменения адресов...
;*******************************************************
; Поместить адреса обработчиков прерываний в таблицу.
; вход: es:di - указывает на начало таблицы.
;*******************************************************
GetVectors      proc    near
                sub     ax,ax
                mov     ds,ax           ; ds = 0...
                PutIntoTable    8h,0
                PutIntoTable    13h,4
                PutIntoTable    25h,8
                PutIntoTable    26h,12
                PutIntoTable    21h,16
                ret
GetVectors      endp
;*******************************************************
; Восстановить первоначальные адреса обработчиков прерываний.
;*******************************************************
RestoreVectors  proc    near
                sub     ax,ax
                mov     ds,ax           ;ds = 0...
                MoveSeg es,cs
                lea     di,es:OldVectorTable ;es:di указывает на начало таблицы
                TakeOutTable    8h,0 ;адресов обработчиков прерываний..
                TakeOutTable    13h,4
                TakeOutTable    25h,8
                TakeOutTable    26h,12
                TakeOutTable    21h,16
                ret
RestoreVectors  endp
;*******************************************************
; Определить имя выполняемой в данный момент фоновой программы.
; выход: es:dx - указывает на начало имени программы,
;               cx - длина имени программы.
;*******************************************************
FindName        proc    near
                mov     ah,62h
                int     21h ;Получить сегментную компоненту адреса блока PSP
                mov     es,bx ;в es...
                mov     es,es:[2ch] ;es = начало среды программы...
                cld
                mov     di,0
                mov     cx,7fffh ;cx = максимальный размер среды...
lp:             cmp     byte ptr es:[di],0
                jz      cmd_nam ;Пропустить set переменную
                xor     al,al ;пока не найдено два нулевых байта подряд...
                repne   scasb
                cmp     cx,0
                jnz     lp
error:          MoveSeg es,cs ;Имя не найдено...
                lea     dx,ErrMsg ;es:dx указывает на ErrMsg...
                mov     cx,ErrMsgLen
                ret
cmd_nam:        inc     di ;Найдено два нуля подряд...
                cmp     word ptr es:[di],1;Теперь должно быть слово, равное 1...
                jnz     error ;Если нет слова, то ошибка...
                add     di,2
                mov     dx,di ;es:dx = указатель на начало имени программы...
                xor     al,al ;al = 0 - признак конца имени программы...
                mov     cx,-1
                repne   scasb ;Искать конец имени программы...
                mov     cx,di ;cx = указатель на конец имени программы...
                sub     cx,dx
                dec     cx ;cx = длина имени программы...
                ret
FindName        endp
;*******************************************************************
; Сохранить размер курсора и координаты его положения на экране.
;*******************************************************************
SaveCursor      proc    near
                PushReg <ax,dx,cx>
                Video   0f ;Получить номер текущей видеостраницы...
                Video   3 ;Получить размер и положение курсора...
                mov     cs:OldCursorShape,cx
                mov     cs:OldCursorPos,dx
                mov     cx,2000h
                Video   1 ;Удалить курсор...
                mov     bl,CurrentAtr ;Установить атрибут для вывода на экран...
                PopReg  <cx,dx,ax>
                ret
SaveCursor      endp
;*******************************************************************
; Восстановить размер курсора и его положение на экране.
;*******************************************************************
RestoreCursor   proc    near
                PushReg <ax,cx,dx>
                Video   0f ;Получить номер текущей видеостраницы...
                mov     cx,cs:OldCursorShape
                Video   1 ;Восстановить размер курсора...
                mov     dx,cs:OldCursorPos
                Video   2 ;Восстановить положение курсора...
                PopReg  <dx,cx,ax>
                ret
RestoreCursor   endp
;*******************************************
; Ждать нажатия клавиши и получить ее код.
;*******************************************
GetKey          proc    near
                mov     ah,01h
                int     16h ;Ждать нажатия клавиши...
                jz      GetKey
                mov     ah,0
                int     16h ;Получить ее код...
                ret
GetKey          endp
;*******************************************************************
; Выполнить операцию с участком экрана.
; вход: dx - Координаты верхнего левого угла участка экрана
;            ( dh-строка, dl-колонка ),
;       cx - Размер участка экрана
;            ( ch - по строкам, cl - по колонкам ),
;       ds:si - Указатель на буфер сохранения/восстановления или
;            содержимое памяти, которое надо вывести на экран.
;*******************************************************************
BoxHandler      proc    near
                push    bx
                add     cx,dx ;Вычислить конечные координаты участка...
next_string:    push    dx
next_column:    push    cx
                Video   2 ;Установить курсор в позицию, определяемую dx...
                cmp     di,InversFunction
                jz      invers_box ;Поменять местами цвета фона и символов ?
                cmp     di,ShadeFunction
                jz      make_shade ;Нарисовать тень окна ?
                cmp     di,PutFunction
                jz      put_box ;Вывести что-нибудь на экран ?
                cmp     di,RestoreFunction
                jz      restore_box ;Восстановить участок экрана ?
;Сохранить участок экрана...
                Video   8 ;Прочитать символ и атрибут...
                mov     [si],ax ;Сохранить символ и атрибут в буфере...
                add     si,2
                jmp     further
restore_box:    mov     ax,[si] ;ax = символ и атрибут из буфера...
                mov     bl,ah ;bl = атрибут...
                mov     cx,1 ;Установить число выводимых символов в 1...
                Video   9 ;Вывести символ и атрибут...
                add     si,2
                jmp     further
put_box:        mov     al,[si] ;bl = CurrentAtr...
                mov     cx,1 ;Установить число выводимых символов в 1...
                Video   9 ;Вывести символ и атрибут...
                inc     si
                jmp     further
invers_box:     Video   8 ;Прочитать символ и атрибут...
                mov     bl,ah ;bl = атрибут...
                mov     cl,4 ;Поменять местами старшие и младшие 4 бита...
                shr     bl,cl
                shl     ah,cl
                or      bl,ah
                mov     cx,1 ;Установить число выводимых символов в 1...
                Video   9 ;Вывести символ и атрибут...
                jmp     further
make_shade:     Video   8 ;Прочитать символ и атрибут...
                mov     bl,ShadeAtr ;bl = атрибут тени...
```

```
                mov   cx,1 ;Установить число выводимых символов в 1...
                Video     9 ;Вывести символ и атрибут...
                inc   si
further:        pop   cx
                inc   dl ;Увеличить число столбцов...
                cmp   dl,cl ;Достигнут заданный предел по столбцам ?
                jl    next_column ;Нет..
                pop   dx
                inc   dh ;Увеличить число строк...
                cmp   dh,ch ;Достигнут заданный предел по строкам ?
                jl    next_string ;Нет..
                pop   bx
                ret
BoxHandler      endp
;*********************************************
;  Меню - "Да - Нет".
;  выход: флаг CARRY установлен, если "Нет".
;*********************************************
YesOrNo         proc      near
                push      bp
                MoveSeg  ds,cs
                Box   si,NoPos,NoLen,InversFunction ;Выбрать пункт меню "Нет"...
                mov   bp,0FFFh   ;bp = флаг выбора...
get_next_key:   call  GetKey   ;Получить код клавиши...
                cmp   al,EnterKey  ;Это возврат каретки ?
                jz    exit_yes_or_no    ;Да...
                Box   si,NoPos,NoLen,InversFunction ;Иначе сменить пункт меню...
                Box   si,YesPos,YesLen,InversFunction
                not   bp  ;Инвертировать флаг выбора...
                jmp   get_next_key
exit_yes_or_no: mov   ax,bp
                sahf  ;Сохранить выбранную пункт меню в регистре флагов...
                pop   bp
                ret
YesOrNo         endp
;*****************************************************************
;  Сформировать на экране вопрос о переназначении вектора.
;  вход: cx - смещение в таблице адресов обработчиков прерываний,
;                 в котором обнаружено различие между старым и новым
;                 значениями адреса.
;*****************************************************************
VectorQuestion  proc      near
                push      cx
                MoveSeg  ds,cs
                Box   VecMsg,VecMsgPos,VecMsgLen,PutFunction
                pop   cx
                cmp   cx,8  ;Это попытка переназначить вектор 8h ?
                jl    it_is_vector_13
                lea   si,Vec8Msg
                jmp   vector_found
it_is_vector_13:  cmp   cx,6  ;Это попытка переназначить вектор 13h ?
                jl    it_is_vector_25
                lea   si,Vec13Msg
                jmp   vector_found
it_is_vector_25:  cmp   cx,4  ;Это попытка переназначить вектор 25h ?
                jl    it_is_vector_26
                lea   si,Vec25Msg
                jmp   vector_found
it_is_vector_26:  cmp   cx,2  ;Это попытка переназначить вектор 26h ?
                jl    it_is_vector_21
                lea   si,Vec26Msg
                jmp   vector_found
it_is_vector_21:  lea   si,Vec21Msg  ;Это попытка переназначить вектор 21h...
;Вывести соответствующий номер вектора в окно...
vector_found:   Box   si,VecXMsgPos,VecXMsgLen,PutFunction
                ret
VectorQuestion  endp
;****************
;  Открыть окно.
;****************
PutWindow       proc      near
                PushReg   <ax,dx,cx,di,si,ds,es>
                MoveSeg  ds,cs
                mov   cx,WindowSize   ;cx = размер окна...
                add   cx,0102h   ;cx = размер окна с поправкой на тень...
;Сохранить участок экрана...
                Box    SaveBuffer,WindowPos,cx,SaveFunction
                mov   dx,WindowPos
                add   dx,0102h
;Вывести прямоугольник, имитирующий тень...
                Box    si,dx,WindowSize,ShadeFunction
;Вывести окно...
                Box    Window,WindowPos,WindowSize,PutFunction
                PopReg    <es,ds,si,di,cx,dx,ax>
                ret
PutWindow       endp
;*********************************************
;  Закрыть окно и восстановить участок экрана.
;*********************************************
RestoreBox      proc      near
                PushReg   <ax,dx,cx,di,si>
                mov   cx,WindowSize   ;cx = размер окна...
                add   cx,0102h   ;cx = размер с поправкой на тень...
;Восстановить участок экрана...
                Box   SaveBuffer,WindowPos,cx,RestoreFunction
                PopReg    <si,di,cx,dx,ax>
                ret
RestoreBox      endp
;*********************************************
;  Вывести строку в середину окна.
;  вход: ds:si - указатель на строку,
;           ax - длина строки.
;*********************************************
TypeString      proc      near
                mov   dx,WindowSize
                sub   dl,al  ;dl = (размер окна по колонкам - длина строки)...
                mov   cl,1
                shr   dl,cl  ;dl = dl / 2 ...
                mov   cx,WindowPos  ;dl = номер колонки на экране...
                add   dl,cl
                mov   cx,FileMsgPos
                mov   dh,ch  ;Вывести строку с именем под  строкой "FileMsg"...
                inc   dh
                mov   cx,ax   ;cx = длина строки...
                Box   si,dx,cx,PutFunction
                ret
TypeString      endp
;*********************************************
;  Вывести вопрос: "Вы хотите открыть файл ?".
;*********************************************
FileQuestion    proc      near
                PushReg   <dx,ax,ds>
                MoveSeg  ds,cs
                Box   FileMsg,FileMsgPos,FileMsgLen,PutFunction
                PopReg    <ds,ax,si>
                call  TypeString
                ret
FileQuestion    endp
;*****************************************************************
;  Вывести вопрос: "Вы хотите оставить программу резидентной ?".
;*****************************************************************
MemoryQuestion  proc      near
                PushReg   <dx,cx,es>
                MoveSeg  ds,cs
                Box   MemMsg,MemMsgPos,MemMsgLen,PutFunction
                PopReg    <ds,ax,si>
                call  TypeString
                ret
                MemoryQuestion    endp
;*****************************************************************
;  Открыть окно. вывести вопрос, и организовать меню "Да - Нет".
;  вход: di - флаг вопроса ( о чем спрашивать ?).
;*****************************************************************
AskUser         proc      near
                call  SaveCursor  ;Сохранить позицию и размер курсора...
                call  PutWindow  ;Открыть окно...
                cmp   di,AskAboutFile
                jz    file_question  ;Задать вопрос о файле ?
                cmp   di,AskAboutVector
                jz    vector_question  ;Задать вопрос о секторе ?
                cmp   di,AskAboutMemory
                jz    memory_question  ;Задать вопрос о памяти ?
MoveSeg         ds,cs
;Надо спросить и секторе...
                Box   SecMsg,SecMsgPos,SecMsgLen,PutFunction
                jmp   answer
file_question:  call  FileQuestion
                jmp   answer
vector_question:  call  VectorQuestion
                jmp   answer
memory_question:  call  MemoryQuestion
answer:         call  YesOrNo  ;Меню "Да-Нет"...
                pushf      ;Сохранить регистр флагов...
                call  RestoreBox  ;Стереть окно...
                call  RestoreCursor  ;Восстановить курсор...
                popf       ;Восстановить регистр флагов для использования...
                ret
AskUser         endp
;
Copyright       db    10,13
                db    "****************************************************",10,13
                db    "*              Virus Protector V1.0                *",10,13
                db    "****************************************************",10,13
                db    "*            Written by V.Komyagin                 *",10,13
                db    "****************************************************",10,13,'$'
AlreadyMsg      db    10,13,"Virus Protector уже загружен...",10,10,13,'$'
;Начало работы программы...
init:           MoveSeg  ds,cs
                mov   ax,3508h  ;Запросить в es:bx адрес обработчика
                int   21h       ;прерывания от таймера...
                mov   ax,es:[bx-2]    ;Выше адреса обработчика находится
                cmp   ax,VP_sign      ;слово VP_sign ?
                jnz   not_load        ;Нет...
                lea   dx,AlreadyMsg   ;Да, VP уже загружен...
                mov   ah,09h          ;Вывести сообщение об этом
                int   21h             ;и окончить работу...
                mov   ax,4c01h
                int   21h
not_load:       mov   old8o,bx  ;Сохранить адрес обработчика прерывания,
                mov   old8s,es  ;сформировав инструкцию JMP FAR old8s:old8o...
                lea   dx,Copyright  ;Вывести заголовок...
                mov   ah,09h
                int   21h
;Переназначить вектора на себя...
                ChangeHandler     21
                ChangeHandler     27
                ChangeHandler     26
                ChangeHandler     13
                MoveSeg  es,cs
;Сохранить адреса обработчиков прерываний...
                lea   di,es:OldVectorTable
                call  GetVectors
                push      cs
                pop   ds
                lea   dx,New8
                mov   es:[di],dx   ;Поместить в таблицу новый адрес для
                mov   es:[di+2],cs  ;8-го вектора...
                lea   dx,New8  ;Переназначить 8-й вектор на себя...
                mov   ax,2508h
                int   21h
                lea   dx,Copyright  ;Все, что выше метки Copyright, останется
                mov   cl,4  ;резидентно в памяти...
                shr   dx,cl  ;Для этого вычислим размер памяти в параграфах...
                add   dx,11h  ;Прибавим размер блока PSP...
                mov   ax,3100h ;Завершим выполнение и оставим резидентную
                jmp   old21  ;часть. Сделаем так, чтобы VP не поймал себя...
Code            Ends
End             Start
```

ВНИМАНИЮ СПЕЦИАЛИСТОВ ПО МАШИННОЙ ГРАФИКЕ!

CAD und Computergraphik

"Интеркомпьютер" и Австрийская ассоциация компьютерной графики (ACGA) планируют издать в 1991 г. специальный выпуск журнала "CAD und Computergraphik", составленный из статей советских авторов.
Журнал "CAD und Computergraphik" издается в Австрии на немецком языке и распространяется в Австрии, Германии и Швейцарии.

Выход в свет специального выпуска приурочен к открытию 12-й Международной конференции по машинной графике - "EUROGRAPHICS'91", которая будет проводиться в Вене со 2 по 6 сентября 1991 г.

Тематика статей для специального выпуска журнала "CAD und Computergraphik":

- **синтез изображений и анимация;**
- **технические средства машинной графики;**
- **взаимодействие человек-компьютер;**
- **алгоритмы визуализации;**
- **моделирование;**
- **графические стандарты;**
- **обработка изображений;**
- **применение машинной графики.**

Объем статей - до 0,8 авторского листа (30 000 знаков).
Отбор статей осуществляется международным редакционным советом на конкурсной основе.
Авторы статей, опубликованных в специальном выпуске журнала "CAD und Computergraphik", будут приглашены на конференцию "EUROGRAPHICS'91".
Просим присылать статьи в редакцию "Интеркомпьютера" до 1 марта 1991 г.



КОПИЛКА ОПЫТА

Резидентные программы на языке Си. Рекомендации разработчикам

В.Э. ДЕМБСКИЙ

Для организации псевдомультизадачного режима в операционной системе MS-DOS применяются резидентные (постоянно находящиеся в оперативной памяти) программы, которые активизируются при возникновении прерываний.

Резидентные программы для MS-DOS, как правило, пишутся на языке ассемблера. Такие программы занимают мало места в оперативной памяти и работают быстро. Однако современные компиляторы языков высокого уровня позволяют писать резидентные программы, мало уступающие программам, написанным на языке ассемблера. Одним из компиляторов, позволяющих создавать резидентные программы, является оптимизирующий компилятор Microsoft C, версия 5.1.

Если программа должна быть резидентной, то при своем завершении она обращается к функции 31H операционной системы MS-DOS, известной под названием Terminate & Stay Resident (завершить работу и остаться резидентной). При программировании на языке Си вызов этой функции осуществляется функцией _dos_keep (rc,size), где rc - код завершения, а size - размер резидентной части программы в блоках, длина каждого из которых равна 16 байт. Определение этого размера при программировании на языке Си имеет свои особенности.

Распределение памяти под программу в MS-DOS в общем случае выглядит следующим образом:

Старшие адреса памяти

Область памяти	Граница
Свободный объем оперативной памяти для динамического размещения данных	
	- Конец программы (начало стека)
Сегмент стека	
	- Конец стека
Сегмент данных	
Сегмент кода	
	- Начало программы (начало кода)

Младшие адреса памяти

Перед нами встает задача определения размера программы, т.е. разности между концом сегмента стека или концом сегмента данных и началом сегмента кода, так как сегмент стека используется только при загрузке и инициализации программы.

Адрес начала сегмента кода определить легко, его можно положить равным адресу специального блока PSP - Program Segment Prefix (префикс сегмента программы). Компилятор Microsoft C, версия 5.1, инициализирует специальную глобальную переменную с именем _psp, которая содержит в себе сегментную компоненту адреса блока PSP.

Адрес конца сегмента стека определить труднее. Мы предлагаем использовать для этого глобальную переменную atopsp, которая в документации по данному компилятору не описана; назначение ее можно уяснить на основе анализа файла crt0.asm (исходные тексты модулей инициализации поставляются вместе с компилятором). В эту переменную при загрузке программы заносится содержимое регистра SP (смещение адреса конца стека). Содержимое регистра SS (сегментную компоненту этого адреса) можно определить с помощью функции segread (&sr).

Если в программе предусмотрено динамическое размещение данных, то к размеру ее резидентной части следует добавить размер области оперативной памяти, необходимой для этого. Для эффективного использования этой памяти необходимо задать значение глобальной переменной _amblksiz, равное размеру требуемой области.

В заключение напомним, что операционная система MS-DOS нереентерабельна, и обработчики прерываний не могут вызывать никаких функций DOS, но можно использовать базовую систему ввода-вывода - BIOS. Не забудьте в описании функций, используемых в качестве обработчиков прерываний, указать описатель interrupt, который сообщает компилятору о том, что данная функция работает как обработчик прерываний. Перед завершением работы программы можно освободить блок PSP, если резидентная часть программы не обращается к его содержимому.

Предложенный метод дает возможность точно определить размер резидентной части программы.

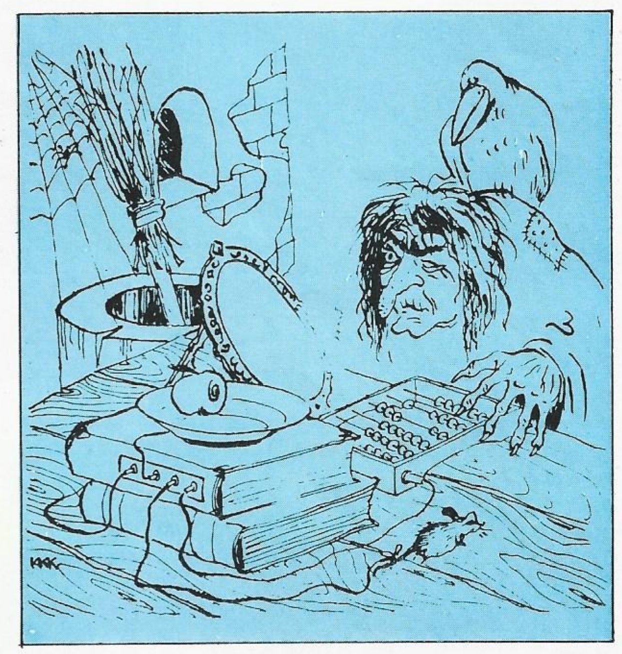
Ниже приведен пример программы, реализующей этот метод:

```
#include <stdio.h>
#include <stdlib.h>
#include <alloc.h>
void main ( int argc, char **argv )
{
/*
* Описатель Huge упрощает работу
* компилятора при выполнении
* вычислений с указателями
*/
    char huge *tsrstack;
    char huge *tsrbottom;
    unsigned tsrsize;
    struct SREGS sr;
    extern unsigned _atopsp;
        segread ( &sr );
        FP_OFF ( tsrstack ) = _atopsp;
        FP_SEG ( tsrstack ) = sr.ss;
        FP_SEG ( tsrbottom ) = _psp;
        FP_OFF ( tsrbottom ) = 0;
        _amblksiz = 512;
/*
* Размер программы в блоках по 16 байт:
* начало стека
* конец программы
* память для динамического размещения
* данных
*/
    tsrsize = ( ( tsrstack -
        tsrbottom ) >> 4 )
        + ( _amblksiz >> 4 );
/*
* Основная часть инициализации
* программы:
* установка обработчиков прерываний
* анализ параметров и др.
*/
/*
* Освобождение блока PSP и
* завершение работы с оставлением рези-
* дентной части программы в оперативной
* памяти
*/
    _dos_freemem( _psp );
    _dos_keep( 0, tsrsize );
}
```

ФАЙЛ ха.ха

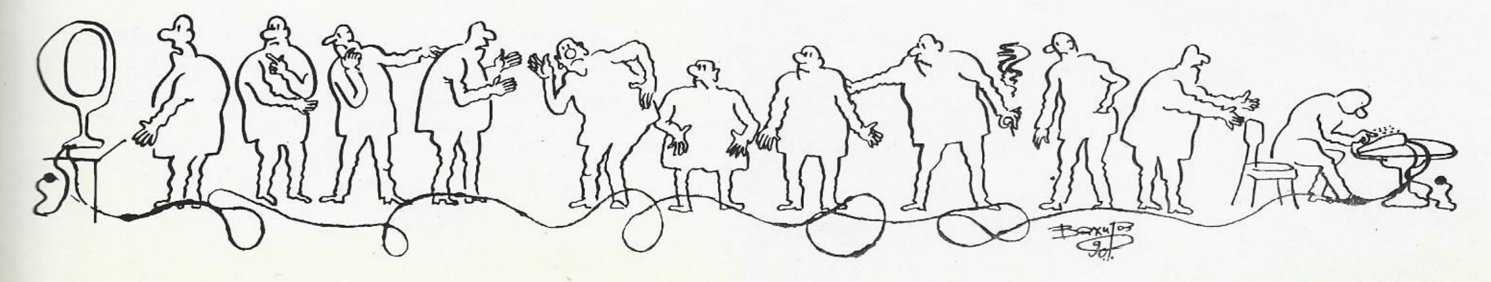
ИЗ РИСУНКОВ, ПРИСЛАННЫХ НА НАШ КОНКУРС







Авторы: Ю.Л. МЕШАЛКИН
А.А. ПОЖАЛОВ
О.В. КЛЮКИН
Ф.А. ВАХИТОВ



ФЛОРЕНТИЙСКОЕ ИСКУССТВО В КОМПЬЮТЕРЕ

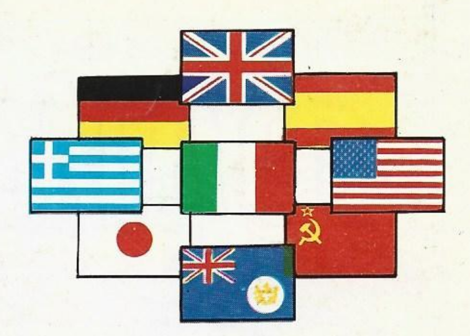




ОТ ОТДЕЛЬНЫХ КОМПЬЮТЕРОВ ДО СЛОЖНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ!

Решить эту и другие проблемы в кратчайшие сроки вам помогут специалисты СП "АНТАРЕКС С.У."

НАШ АДРЕС:
117342, Москва, ул. Обручева, 36
Телефон: 334-29-74 Телефакс: 4202250
Телекс: 412160 ANTAR SU



HANTAREX

Совместное советско-итальянское предприятие "HANTAREX S.U." ("АНТАРЕКС С.У."), иностранным учредителем которого является фирма "HANTAREX S.p.A." (штаб-квартира во Флоренции, Италия, отделения в Японии, США, Греции, ФРГ, Испании, Великобритании, Гонконге) - одна из немногих европейских фирм, производящих персональные компьютеры, мониторы и другое электронное оборудование, специализируется на решении всевозможных задач компьютеризации.

СП "АНТАРЕКС С.У." ПРЕДЛАГАЕТ:

- ☐ широкий спектр персональных компьютеров, рабочих станций и файл-серверов;
- ☐ локальные сети;
- ☐ программное обеспечение (от MS-DOS и любых пакетов для нее до SCO UNIX/XENIX System V/286/386 с различными пакетами для них, в том числе SCO Microsoft WORD 5.0, AutoCAD, и разнообразных САПР);
- ☐ аппаратно-программные комплексы для медицинских приложений;
- ☐ сопутствующее оборудование (оргтехника, цифровые телефонные станции, устройства бесперебойного питания, телефаксы, видеотехника, приемные устройства спутниковых систем телевидения, телевизионные "видеостены").

К ВАШИМ УСЛУГАМ:

- ☐ русификация компьютеров и принтеров;
- ☐ установка, ввод в эксплуатацию;
- ☐ гарантийное и послегарантийное обслуживание;
- ☐ разработка и адаптация программного обеспечения.

Поставка оборудования, имеющего стандартную конфигурацию, в двухнедельный срок.

Гарантия 2 года, обслуживание в любом регионе СССР и за его пределами.

СП "АНТАРЕКС С.У." ЖДЕТ ВАС!